DISEÑO Y ANÁLISIS COMPARATIVO DEL ALGORITMO HIBRIDO ANT COLONY- EVOLUTIVO CON RESPECTO AL ALGORITMO ANT COLONY EN LA SOLUCIÓN DE UN PROBLEMA DE OPTIMIZACIÓN MULTIOBJETIVO EN REDES OPTICAS

CARLOS JULIO ARDILA HERNÁNDEZ

UNIVERSIDAD DEL NORTE
DIVISIÓN DE INGENIERÍAS
MAESTRÍA EN INGENIERÍA INDUSTRIAL
BARRANQUILLA
2006

DISEÑO Y ANÁLISIS COMPARATIVO DEL ALGORITMO HIBRIDO ANT COLONY- EVOLUTIVO CON RESPECTO AL ALGORITMO ANT COLONY EN LA SOLUCIÓN DE UN PROBLEMA DE OPTIMIZACIÓN MULTIOBJETIVO EN REDES OPTICAS

CARLOS JULIO ARDILA HERNÁNDEZ

MONOGRAFÍA DE GRADO

Presentado como requisito para optar al título de Magíster en Ingeniería Industrial

Director

Ph.D Ing. Yesid Donoso Meisel

UNIVERSIDAD DEL NORTE

DIVISIÓN DE INGENIERÍAS

DEPARTAMENTO DE INGENIERÍA DE SISTEMAS

BARRANQUILLA

2006

Nota de Aceptación:

_____
_____
_____
_____
_____
_____

_____
Pd. D. Ing. Yesid Donoso Meisel
Director del Proyecto

_____
Ph. D. Ing. Carlos Paternina A.
Coordinador Programa Maestría
Ingeniería Industrial.

_____
Corrector

_____
Jurado

_____
Jurado

Barranquilla, 30 de Mayo de 2006

"A mi esposa Astrid y a mis hijos Daniela y Julián , por darme la energía que me motiva a mejorar cada día como persona, a comprender que lo tenemos todo, pero a la vez no tenemos nada, y que sólo el amor, la lealtad y el trabajo honesto, nos permiten dejar una huella imborrable en la vida".

AGRADECIMIENTOS

Con el presente trabajo, quiero agradecer a mi amigo y compañero de labores Ph. D., Ingeniero Yesid Donoso Meisel por todo su apoyo, sus sugerencias y consejos que me permitieron conocer y trabajar en tan interesante temática.

A mi discípulo y amigo Ingeniero de Sistemas Alfredo José Pérez Martínez, por toda su generosa e invaluable colaboración.

A mis compañeros de trabajo por escucharme y motivarme a continuar.

A los profesores, Ing. Rodrigo Barbosa Correa Ph. D., Ms.C. Martín Díaz y al Ing. Alfonso Mancilla Herrera Ms. C., por todas las sugerencias dadas en el aspecto estadístico.

A mi amigo y compañero de estudio Ing. Rodrigo Wadnipar, por su estimulo y solidaridad.

RESUMEN

La masificación del uso de Internet por parte de empresas y personas naturales ha creado una ampliada variedad de tráfico en la red. Nuevos servicios sobre Internet tales como video por demanda, televisión, tele-conferencias y telefonía, entre otros, requieren cada vez más recursos. La Segunda Generación de la Internet Óptica (SGOI) representa una solución a los problemas de demanda de recursos en Internet, así como una aproximación más eficiente para proporcionar servicios IP en la parte alta de la capa óptica mediante GMPLS.

En la presente investigación se plantea y soluciona un Modelo de Optimización Multiobjetivo (MOP) para la transmisión multicast en redes ópticas, propuesto por el Ing. Yesid Donoso Meisel Ph. D., en donde se considerara la minimización de 11 funciones objetivos, las cuales involucran las siguientes variables de decisión: atenuación máxima en el enlace de la fibra óptica, retardo total, el retardo máximo, el retardo promedio, máxima variación del retardo, el número de longitudes de onda utilizadas, número de saltos total, el retardo promedio, el retardo máximo, máxima variación del retardo y ancho de banda, El modelo fue resuelto aplicando la metaheurística Ant Colony (Colonias de Hormigas) y un nuevo algoritmo de concepción hibrida en el cual se aplica al algoritmo Ant Colony, asignación de fitness, selección ambiental y cruzamiento propios de los algoritmos evolutivos, estos procedimientos se obtuvieron del algoritmo evolutivo Strength Pareto Evolutionary Algorithm 2 (SPEA2).

De acuerdo a los resultados obtenidos al aplicar los dos algoritmos en las topologías de redes NSF, MCI y Sprint, las métricas Generación de Vectores No Dominados (GVND), Distancia Generacional (DG) y Spacing, el algoritmo Ant Colony Evolutivo tuvo un mejor desempeño que el algoritmo Ant Colony al producir soluciones en promedio menores que el algoritmo Ant Colony. En lo que respecta al tiempo de ejecución se encontró que para el problema el algoritmo Ant Colony arrojó mejores promedios.

TABLA DE CONTENIDO

LISTA I ix GURAS

Página

LISTA DE TABLAS

xi

1 INTRODUCCIÓN

1.1 PLANTEAMIENTO DEL PROBLEMA

En la Ingeniería, al igual que en muchos problemas de la vida cotidiana, es muy frecuente encontrar problemas que implican la maximización o minimización de alguna función o recurso, disciplina matemática que se conoce con el nombre de optimización. Mas aún, existen muchos problemas cuya solución implica maximizar y/o minimizar varias funciones al mismo tiempo, lo cual conduce a que, en ocasiones, el mejoramiento de una función produzca el desmejoramiento de otra.

Ante la presencia de problemas de optimización con varios objetivos, la perspectiva o concepción de lo que significa un óptimo, cambia con respecto a los problemas de optimización que presentan un solo objetivo, ya que en la optimización mono-objetivo la solución al problema es única, mientras que en un problema multi-objetivo la solución no necesariamente es única, sino, que podría ser un conjunto de soluciones.

Por otra parte, con el auge de Internet, actualmente a través de esta red aumentan cada vez más los servicios a los usuarios, por ejemplo, servicios de televisión y radio, videoconferencias, etc. Estos servicios, consumen y disputan recursos de la red, tales como un mayor ancho de banda, tiempos de respuestas menores, lo que implica optimizar éstos y otros recursos sobre la red, con el fin de asignar a cada servicio los recursos necesarios y lograr con ello un desempeño óptimo de la red al prestar los múltiples servicios convirtiendo la utilización de aplicaciones Web atractivas para el usuario final.

Con el actual incremento del volumen de tráfico, y con la introducción de nuevas aplicaciones de tiempo real, multimedia y de multicast, los protocolos y servicios tradicionales de Internet de primera generación son inadecuados. La Segunda Generación de la Internet Óptica (SGOI) representa una solución a los problemas de demanda de recursos en Internet, así como una aproximación más eficiente para proporcionar servicios IP en la parte alta de la capa óptica mediante GMPLS y unida a estos avances tecnológicos,

surgen además investigaciones, que buscan nuevas alternativas para ofrecer esos anchos de banda y manejo de calidad y servicio (QoS), por medio de los últimos avances de tecnología óptica. El presente trabajo de investigación, surge de las necesidades anteriormente expuestas y en ella se plantea una nueva alternativa al diseñar un algoritmo de concepción híbrida, que permita optimizar sobre una red de fibra óptica con transmisión multicast el desempeño de una red expresada mediante un modelo matemático con 11 funciones objetivos.

1.2 OBJETIVOS

Objetivo General

- Diseñar y comparar un algoritmo de característica híbrida, que permita la solución de un problema de optimización multiobjetivo en redes ópticas para la transmisión multicast.

Objetivos Específicos

- Diseñar el algoritmo híbrido Ant Colony Evolutivo, tomando como base la metaheurística Ant Colony y el algoritmo evolutivo Strength Pareto Evolutionary Algorithm 2 (SPEA2).

- Comparar la eficiencia del algoritmo mediante la comparación de los resultados obtenidos al aplicar el algoritmo híbrido y la metaheurística conocida Ant Colony.

1.3 CONTRIBUCIONES

A continuación se listan las contribuciones del presente trabajo de investigación:

- Se diseñó un nuevo algoritmo de naturaleza híbrida tomando el algoritmo Ant-Colony, e implementando en él, procedimientos propios del algoritmo evolutivo SPEA II tales como asignación de fitness, selección ambiental, selección y cruzamiento.

- Aplicación del algoritmo híbrido en la solución un problema multi-objetivo con transmisión multicast en árboles estáticos Origen-Destino, sobre redes GMPLS.

- El problema multiobjetivo planteado, se solucionó mediante la metaheurística diseñada, resolviendo un problema NP-Hard en tiempo polinomial.

1.5 ANTECEDENTES

La presente de investigación no es el producto del azar, sino por el contrario es el producto de otra serie de trabajos entre los que cabe resaltar la tesis doctoral realizada por el Ing. Yesid Donoso Meisel, titulada "Multi-Objective Optimization Écheme for Static and Dynamic Multicast Flor", y algunas tesis de pregrado dirigidas algunas por el Ing. Yesid Donoso y otras por este autor, en la que es preciso señalar: "Análisis comparativo de los algoritmos evolutivos NSGA-II y SPEA-II mediante la solución de un problema de optimización multiobjetivo", realizada por Carolina Alvarado e Ivan Herazo, "Optimización multi-objetivo en transmisiones multicast sobre redes ópticas usando algoritmos evolutivos, meméticos y ant-colony ", realizada por Luis Caro y Pier Rosado y "Optimización multiobjetivo en redes multicast mediante algoritmos , meméticos" , realizada por Alfredo Pérez.

Por otra parte la exploración bibliografica sobre trabajos similares como es la combinación de metaheurística para resolver problemas multiojetivos de transmisión multicast en fibra óptica arrojó resultados nulos, tan solo se encontraron trabajos interesantes en otras áreas tales como:

- “Ant direction hybrid differential evolution for solving large capacitor placement problems”, realizado por JP Chiou y CT Su, articulo escrito para IEEE TRANSACTION ON POWER SYSTEM, el cual presenta un hibrido de una hormiga diferencial con evolución (ADHE), con programación entera que es efectiva y eficiente para resolver problemas en sistemas distribuidos de colocación de capacitares grandes, utilizando la combinación de las metaheurísticas Ant Colony y Simulated Annealing.

- “ A hybrid metaheuristic for the cuadratic assignment problem”, realizado por LY Tseng y SC Liang, articulo escrito para COMUTACIONAL OPTIMIZATION AND APPLICATION, el cual presenta una metaheurística hibrida llamada ANGEL que combina la optimización Ant Colony (ACO) con algoritmo genético (GA) y un método de búsqueda local (LS), para la solución del problema en la fase de GA en vez de comenzar con una población inicial que consiste de cromosomas generados aleatoriamente, el GA tiene una población inicial construída por ACO para así empezar el proceso con un buen inicio.

1.4 ORGANIZACIÓN DEL DOCUMENTO

En el capítulo 2, se presentan los conceptos básicos concernientes a la optimización multiobjetivo; se desarrolla la teoría fundamental de la metaheurística Ant-Colony, algoritmos evolutivos (SPEA2) y la teoría sobre redes ópticas.

Seguidamente, en el capítulo 3, se describe el diseño del algoritmo híbrido Ant Colony Evolutivo, su complejidad y se presenta y explican cada una de las funciones y

restricciones que conforman el modelo matemático sobre el cual se experimentará el algoritmo.

En el capítulo 4, se realiza el respectivo análisis comparativo de resultados, producto de las soluciones generadas por el Ant Colony (MOANTCOL) y el Ant Colony Evolutivo (MOANTCOLE). Se explican las métricas de comparación utilizadas: generación de vectores dominados, distancias generacionales, spacing y tiempo computacional.

Por último en el capítulo 5, se presentan las conclusiones de la investigación realizada y se proponen trabajos futuros.

2 MARCO TEÓRICO

Un aspecto diario de la ingeniería es el de enfrentarse a problemas cada vez más complejos, que implican ser resueltos o expresados como problemas de optimización. Algunos de estos problemas son solucionados con técnicas tradicionales de la programación lineal, y otros son de naturaleza no lineal; y aún más, existen problemas con más de un objetivo, tal vez con objetivos contradictorios entre sí, por lo tanto tienen que ser analizados mediante otro paradigma: la Optimización multiobjetivo.

La solución a un problema de optimización mono-objetivo, es un punto, pero en el contexto de la optimización multi-objetivo, la solución podría ser un conjunto de puntos, en el que todos los puntos de la solución tienen la cualidad de ser igualmente óptimos.

El presente capítulo, presenta las definiciones necesarias que fundamentan teóricamente en forma básica y sintetizada el presente trabajo.

2.1 DEFINICIONES

A continuación se definirán los conceptos básicos que hacen parte de la optimización multi-objetivo.

- **El problema de optimización multi-objetivo.** Podría decirse que un problema de optimización multi-objetivo (MOP), es un problema en donde hay que encontrar una solución que implica necesariamente, determinar un vector de variables de decisión, las cuales deben satisfacer las restricciones impuestas en el problema y que además optimizan el vector de funciones, el cual está asociado con las diferentes funciones objetivo. Matemáticamente:

$$\text{“optimizar”}\ \mathbf{z}=\mathbf{f}(\mathbf{x}) \qquad \textbf{(1)}$$

$$\text{Sujeto a}\ \mathbf{x}\ \varepsilon\ \mathbf{X_f} \qquad \textbf{(2)}$$

- **Vector de Decisión.** Se define al vector de decisión $\boldsymbol{x=(x_1,x_2,\ldots x_n)^T}$ como el vector de variables, (también parámetros), que representan las cantidades numéricas cuyos valores deben ser encontrados en un problema de optimización. Las variables pueden ser enteras, reales o una mezcla de ellas.

- **Espacio de Decisión (Espacio de Parámetros).** Conjunto de todos los vectores de decisión (soluciones) de un problema de optimización. Se denota por $\boldsymbol{X}$.

- **Restricciones.** En todo problema de optimización con restricciones, el entorno, la disponibilidad de recursos o las características inherentes al problema, imponen sobre éste un conjunto de restricciones, que deben ser satisfechas al solucionar el problema, con el fin de obtener soluciones factibles.

Las restricciones están dadas en términos de las variables de decisión y de los operadores de (<,<=,=,> y >0). Una posible forma de enunciar restricciones sería:

$$r_i(\mathbf{x})\geq 0;\ \ i=1,2,\ldots..,m \qquad \textbf{(3)}$$

$$s_i(\mathbf{x})\leq 0;\ \ i=1,2,\ldots..,p \qquad \textbf{(4)}$$

$$g_i(\mathbf{x})=0;\ \ i=1,2,\ldots.,k \qquad \textbf{(5)}$$

donde m+p+k debe ser menor que el número de variables de decisión *n*, para que puedan existir grados de libertad a optimizar.

- **Conjunto de Soluciones Factibles.** Este conjunto está conformado por todos los vectores de decisión que satisfacen todas las restricciones de factibilidad., se denota con $\boldsymbol{X_f}$..

- **Función Objetivo.** Denotada con **f**, la función objetivo convierte los vectores de decisión del espacio de decisión en un espacio *K*-dimensional llamado **Espacio Objetivo** (espacio de criterio), $\boldsymbol{Z} \in \mathfrak{R}^{\mathbf{k}}$ ($\mathfrak{R}$, el conjunto de los números reales). **f** es un conjunto de funciones $\mathbf{f=(f_1,f_2\ldots,f_k)^T}$.

- **Vector Objetivo (Vector de criterio, punto).** Denotado con **z** representa la imagen en el espacio objetivo de un vector de decisión a través de **f.**

$$\mathbf{z=f(}x\mathbf{)=(z_1,z_2,..,z_k)=(f_1(}x_1\mathbf{),f_2(}x_2\mathbf{),\ldots,f_k(}x_k\mathbf{))^T}. \qquad \mathbf{(6)}$$

- **Mínimo Global.** Dada la función *f*, supongamos que Θ el conjunto de soluciones factibles es diferente de vacío, entonces **x***, es llamado un mínimo global si para toda $\mathbf{x} \in$ Θ, se tiene que:

$$\mathbf{f(x^*)} < \mathbf{f(x)} \qquad \mathbf{(7)}$$

Al problema de determinar un mínimo global se conoce como el problema de minimización global.

- **Mínimo local.** Dada la función **f**, supongamos que Θ el conjunto de soluciones factibles es diferente de vacío, entonces **x*** , es llamado un mínimo local si para toda $\mathbf{x} \in$ Θ, se tiene que:

$$\mathbf{f(x^*)} \leq \mathbf{f(x)}. \qquad \mathbf{(8)}$$

Las definiciones 2.1.8 y 2.1.9 pueden adaptarse a máximo tan sólo cambiando el sentido de la desigualdad.

- **Relación de Orden por Componente.** Se define la relación de orden por ponente < como: $\mathbf{z^1} < \mathbf{z^2} \Rightarrow z^1_i < z^2_i$, para toda $i = 1\ldots k \wedge \mathbf{z^1} \neq \mathbf{z^2}$, $k \in N^+$.

- **Relación de Orden Débil por Componente.** Se define la relación de orden débil por componente ≤ como: $\mathbf{z^1} \leq \mathbf{z^2} \Rightarrow z^1_i \leq z^2_i$, para toda $i = 1 \ldots k$, $k \in N^+$.

- **Conjunto convexo.** Un conjunto **X** se dice convexo, si para cualquier par de puntos tomados en el conjunto S, la combinación lineal convexa también está o pertenece al conjunto **X**, esto es:

$$\text{Sea } \mathbf{x}_1, \mathbf{x}_2 \in \mathbf{X} \Rightarrow \alpha\, \mathbf{x}_1 + (1-\alpha)\, \mathbf{x}_2. \qquad \textbf{(9)}$$

2.2 OPTIMIZACIÓN DE PARETO

Wilfrido Pareto (1848-1923) propuso una manera de decidir cuándo un vector de decisión era mejor que otro en un problema de optimización multi-objetivo. Matemáticamente, lo que se define para decidir, si un elemento de un conjunto, es mejor que otro elemento del mismo conjunto, es una relación de orden. Cuando se habla de optimización mono-objetivo, la manera de decidir cuándo una solución es mejor que otra, se realiza mediante la relación de orden total de los números reales.

Pareto a través de su concepto de optimalidad, define un orden parcial sobre el conjunto de soluciones factibles y de esta manera existen vectores que son, bajo este orden, comparables ó incomparables entre sí. Cuando se tiene una relación de orden parcial sobre un conjunto, los elementos minimales y maximales pueden no ser únicos, es decir, puede que existan varios elementos mínimos o varios elementos máximos, a continuación se expresan aspectos teóricos respecto a la optimización de Pareto, muchos de los cuales se recopilaron de la referencia[1] .

[1] PEREZ, Alfredo. Optimización multiobjetivo en redes multicast mediante algoritmos meméticos. 177p. Trabajo de Grado (Ingeniero de Sistemas). Universidad del Norte, Departamento de Ingeniería de Sistemas. Barranquilla, Colombia 2005.

- **Optimalidad de Pareto.** Una solución $\boldsymbol{x}^* \in X_f$ es llamada **Óptima de Pareto** si no existe $\boldsymbol{x} \in X_f$ tal que $\mathbf{f}(\boldsymbol{x}) < \mathbf{f}(\boldsymbol{x}^*)$. Si $\boldsymbol{x}^*$ es Óptima de Pareto, $\mathbf{z}^* = \mathbf{f}(\boldsymbol{x}^*)$ es llamada **eficiente.**

- **Conjunto Pareto.** Conjunto de todas las soluciones Óptimas de Pareto $x^* \in X_f$ y se denota por $\boldsymbol{X}^*$.

- **Frente Pareto.** Conjunto de todos los puntos eficientes $\mathbf{z}^* = \mathbf{f}(\boldsymbol{x}^*) \in Z$, donde $\boldsymbol{x}^* \in X^*$ y se denota por $\boldsymbol{Z}^*$.

- **Dominancia de Pareto**[1]. Para dos vectores de decisión cualesquiera $\boldsymbol{x}^1, \boldsymbol{x}^2$

$$\boldsymbol{x}^1 \prec \boldsymbol{x}^2 \ (\boldsymbol{x}^1 \text{ domina a } \boldsymbol{x}^2) \quad \text{si y solo si} \quad \mathbf{f}(\boldsymbol{x}^1) < \mathbf{f}(\boldsymbol{x}^2) \qquad \textbf{(10)}$$

$$\boldsymbol{x}^1 \preceq \boldsymbol{x}^2 \ (\boldsymbol{x}^1 \text{ domina débil a } \boldsymbol{x}^2) \quad \text{si y sólo si} \quad \mathbf{f}(\boldsymbol{x}^1) \leq \mathbf{f}(\boldsymbol{x}^2) \qquad \textbf{(11)}$$

$$\boldsymbol{x}^1 \sim \boldsymbol{x}^2 \ (\boldsymbol{x}^1 \text{ y } \boldsymbol{x}^2 \text{ no son comparables}) \quad \text{si y sólo si} \quad \mathbf{f}(\boldsymbol{x}^1) \not\preceq \mathbf{f}(\boldsymbol{x}^2) \wedge \mathbf{f}(\boldsymbol{x}^2) \not\preceq \mathbf{f}(\boldsymbol{x}^1) \qquad \textbf{(12)}$$

Bajo este enfoque, todas las funciones que hacen parte de un *MOP*, son tomadas como iguales, es decir, ninguna función es más importante que otra. En la figura 1 se puede apreciar lo que se ha dicho anteriormente. A los vectores a,b,c,d,e y f en el espacio factible le corresponden los puntos a',b',c',d',e' y f' respectivamente a través de **f**. En esta figura, **f** es un conjunto de dos funciones, f_1 y f_2.

Los puntos a' y d' son dominados por c', los puntos a', d' y b' son dominados por e' y los puntos d' y b' son dominados por f'. Los puntos c', e' y f', no son dominados por ningún otro punto de la región factible y es por ello que dichos puntos forman, para este caso el Frente Pareto óptimo. De esta forma, los vectores c, e, y f, son vectores no dominados y forman el Conjunto Pareto óptimo.

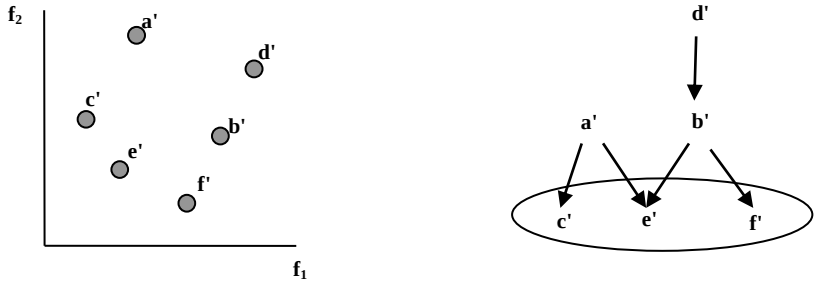


Figura 1

Figura 1. Los cuadrantes dibujados por cada punto en el espacio objetivo de la figura superior, muestran la región del espacio objetivo dominada por cada uno de ellos. En la figura inferior se muestra el diagrama de orden correspondiente, junto con los elementos del Frente de Pareto encerrados en el óvalo.

- **No Dominancia con respecto a un Conjunto.** Se dice que una solución $\boldsymbol{x} \in X_f$ es **no dominada** con respecto al conjunto $X_a \subseteq \boldsymbol{X}_f$ si y sólo si $\exists\, \boldsymbol{x_a} \in X_a, \boldsymbol{x_a} \prec \boldsymbol{x}$.

- **Conjunto No Dominado y Frente no Dominado.** Sea $X_a \subseteq \boldsymbol{X}_f$ y $Z_a = \mathbf{f}(X_a)$. Se define la función $\mathrm{ND}(X_a)$ que retorna el conjunto de soluciones no dominadas de X_a como

$$\mathrm{ND}(X_a) = \{\, \boldsymbol{x_a} \in X_a \mid \boldsymbol{x_a} \text{ es } \textbf{no dominada} \text{ con respecto a } X_a \,\} \quad \textbf{(13)}$$

Similarmente se define la función $\mathrm{ND}(Z_a)$ como:

$$\mathrm{ND}(Z_a) = \{\, \mathbf{z}^i \in Z_a \mid \nexists\, \mathbf{z}^j \in Z_a \text{ tal que } \mathbf{z}^j < \mathbf{z}^i \,\} \quad \textbf{(14)}$$

2.3 MÉTODOS CLASICOS EN LA OPTIMIZACION MULTIOBJETIVO

Los métodos clásicos para optimización multiobjetivo, dan un tratamiento convencional a este tipo de problemas, "convirtiendo" el problema multiobjetivo a tratar en uno monobjetivo, el cual es resuelto mediante métodos conocidos para resolver problemas de este tipo. Existen varios de estos métodos, tales como la suma de pesos; método de la ε - restricción; método de las métricas con pesos y el método de Benson. Los métodos anteriores presentan una desventaja ya que para cada corrida, dichos métodos devuelven sólo un punto del Frente Pareto Óptimo. A continuación se explicarán algunos de estos métodos.

- **Método de la Suma Pesos.** En el método de la suma ponderada, se combinan las funciones objetivos en una sola función, mediante la combinación lineal de las diferentes funciones objetivos, convirtiendo el MOP original en un SOP.

$$\begin{aligned} &\text{Minimizar } y = f(x) = w_1 \cdot f_1(x) + w_2 \cdot f_2(x) + \ldots + w_k \cdot f_k(x) \\ &\text{Sujeto a } \; x \in X_f \end{aligned} \qquad \textbf{(15)}$$

A los coeficientes w_i se les denomina pesos y para cada combinación de los w_i, se encuentra una solución óptima de Pareto. Además, cada w_i son normalizados, es decir,

$$\sum w_i = 1, \; w_j > 0.$$

Bajo la condición de que un algoritmo exacto de optimización es usado y todos los pesos son positivos, este método generará soluciones óptimas de Pareto que pueden ser fácilmente mostradas. La mayor desventaja de esta técnica, es que no puede generar todas las soluciones óptimas de Pareto donde Z_f sea no convexa.

- **Método de la restricción - ε.** En este método, se toma una de las funciones objetivo como función objetivo del nuevo modelo a optimizar y las restantes funciones objetivos se toman como restricciones del modelo, convirtiendo el MOP en un SOP.

$$\begin{aligned} &\text{Minimizar y} = f(x) = f_h(x) \\ &\text{Sujeto a } e_i(x) = f_i(x) \geq \varepsilon_i \\ &x \in X_f \end{aligned} \qquad \textbf{(16)}$$

Los límites inferiores, ε_i, son los parámetros que son variados para encontrar múltiple soluciones óptimas de Pareto.

- **Método Métrica con Pesos.** En este método se establece como función objetivo del nuevo modelo, una combinación de las funciones objetivo del modelo original en la forma:

$$\textbf{(17)} \qquad \begin{aligned} &\text{Minimizar } I_p(x) = \left[\sum_k w_k \left| f_k(x) - z_k^* \right|^p \right]^{1/p} \\ &\text{Sujeto a } x \in X_f \end{aligned}$$

2.4 HEURÍSTICAS Y METAHEURISTICAS

Los métodos clásicos presentan una desventaja y es que ellos dependen de la naturaleza del espacio de soluciones. Cuando el problema multiobjetivo es muy complejo, estos métodos ya no son eficientes para encontrar una solución. Sin embargo, existen alternativas que nos permiten solucionar los problemas de optimización multi-objetivo y en general, un problema de optimización, conocido como heurísticas y *metaheurísticas.*

Con la llegada de la computación se han propuesto algoritmos que encuentran óptimos aproximados, pero dependen mucho del problema que estén tratando. Estos métodos se

conocen con el nombre de Métodos Heurísticos, por ejemplo como nuestro problema a resolver está relacionado con grafos, entonces cabe resaltar los métodos Búsqueda Primero en Profundidad (DFS) y Búsqueda Primero en Amplitud o Anchura (BFS), las cuales son heurísticas que permiten recorrer o visitar de forma eficiente cada vértice y cada arista del grafo exactamente una vez. Sin embargo, si se quiere tratar otra clase de problemas con los mismos algoritmos, no se podría, debido a que estos métodos están hechos para determinada clase de problemas. Esta limitación dio origen al desarrollo de métodos más generales para atacar cualquier problema de optimización; estos métodos son a veces llamados algoritmos de optimización de caja negra (*Black-box Optimization Algorithms*), de optimización general, o simplemente *Metaheurísticas*[2]. Entre las metaheurísticas se encuentran la *Búsqueda Local (Neigborhood Search o Local Search), Recocido Simulado (Simulated Annealing), Búsqueda Tabu (Tabu Search), Algoritmos Genéticos (Genetic Algorithms)* y *Estrategias Evolutivas (Evolutionary Strategies)*[2].

Recientemente, han aparecido dos nuevas metaheurísticas para resolver problemas de optimización, ellas son los *Algoritmos Meméticos* y las *Colonias de Hormigas.*

En general, lo que buscan las metaheurísticas es una exploración "inteligente" del espacio de soluciones. Las metaheurísticas tienen la desventaja de que su resultado son puntos aproximados, es decir, es posible que teóricamente no sean las soluciones óptimas, aunque si se tiene un problema del tipo *NP*, como por ejemplo el problema del *Agente Viajero* o el problema de los *Arboles de Steiner*, el encontrar una muy buena solución es mejor a no tener ninguna.

2.4.1 Método de Búsqueda Primero en Amplitud o Anchura (BFS). La búsqueda Primero en Amplitud (Breadth First Search) es uno de los algoritmos simples para búsqueda en grafos y el arquetipo para algunos algoritmos de grafos importantes. Dijkstra's single source shorest-paths algoritm and Prim's minimun-spanning-tree algoritm son

[2] KNOWLES, Joshua. Local-Search and Hybrid Evolutionary Algorithms For Pareto Optimization.

ejemplos de algoritmos que utilizan una idea similar a la utilizada en la Búsqueda Primero en Amplitud[3].

A continuación mostramos a nivel de pseudocódigo el algoritmo de Búsqueda Primero en Amplitud sobre un grafo simple: **BFS(G, s)**

1 **For** each vertex u ∈ V[G] – {s}
2 **Do** color[u] ← white
3 d[u] ← ∞
4 π[u] ← nil
5 color[s] ← gray
6 d[s] ← 0
7 π[s] ← nil
8 Q ← {s}
9 **while** Q ≠ ϕ
10 **Do** u ← head[Q]
11 **For each** v ∈ Adj[u]
12 **Do if** color[v] = white
13 **Then** color[v] ← gray
14 d[v] ← d[v] + 1
14 π[v] ← u
16 Enqueue(Q, v)
17 Dequeue(Q)
18 color[u] ← black

El procedimiento es el siguiente:

En las instrucciones 1 - 4, se pinta cada vértice de color blanco, el conjunto se hace infinito para cada vértice *u*, y el conjunto de padres de cada vértice se hace vacío. La instrucción 5

[3] BAASE, Sara and VAN GELDER, Allen.Computer Algorithms: Introduction to design and Análisis. Addison Wesley .Massachusetts.2002.

pinta los vértices fuentes *s* de gris, donde éstos son considerados a descubrir cuando el procedimiento empieza. La instrucción 6 inicializa d[s] en 0, y la 7 el conjunto de predecesores de la fuente en vacío. La instrucción 8 inicializa Q la cola que contiene justo los vértices *s*; después de eso Q contiene el conjunto de vértices grises.

El bucle principal de este algoritmo, está contenido en las instrucciones 9 – 18. El bucle itera mientras haya vértices grises, los vértices que no han sido descubiertos todavía, no tienen sus listas de adyacencia examinadas totalmente. La instrucción 10 determina que vértice *u* es la cabeza de la cola Q. El ciclo **for** con las instrucciones 11 – 16 considera cada vértice *v* en la lista de adyacencia de *u*. Si *v* es blanco, entonces no ha sido descubierto y el algoritmo descubre éste mediante la ejecución de las instrucciones 13 – 16. Luego cuando es coloreado de gris, la distancia d[v] es llevada a d[u] + 1. Entonces *u* es grabado como su padre. Finalmente es colocado al final de la cola Q. Cuando sobre la lista de adyacencia de las *u* han sido examinadas, *u* es removido de Q y coloreado de negro en las instrucciones 17 – 18.

2.4.2 Método de búsqueda primero en profundidad (DFS). El método Búsqueda Primero en Profundidad[4] tiene como estrategia, tal como su nombre lo indica, una búsqueda profunda en el grafo siempre que ésta sea posible. Los arcos son explorados fuera del vértice *v* descubierto más recientemente y que aún no ha sido explorado. Cuando todos los arcos de *v* han sido explorados la búsqueda retrocede para explorar arcos del vértice *v* que ha sido descubierto. Este proceso continúa hasta que se hayan descubierto todos los vértices que son alcanzables desde el vértice fuente. Si algunos vértices quedan como no descubiertos, entonces uno de ellos es seleccionado como una nueva fuente y la búsqueda es repetida para esta fuente. Este proceso es repetido hasta que todos los vértices son descubiertos.

Como en la Búsqueda Primero en Profundidad, siempre que un vértice *v* es descubierto, durante una exploración de la lista de adyacencia de un vértice ya descubierto *u*, DFS graba

[4] Ibid 3

este evento ajustando los campos predecesores de los *v*, π[v] a u.. El subgrafo predecesor forma un árbol, el cual puede estar compuesto por diversos árboles, porque la búsqueda puede estar compuesta por diversas fuentes.

El subgrafo predecesor de la Búsqueda Primero en Profundidad, es definido levemente diferente a la Búsqueda Primero en amplitud o anchura: dejamos $G_\pi = (V, E_\pi)$, donde

$E_\pi = \{ (\pi[v], v) : v \in V \wedge \pi[v] = \mathbf{NIL}\}$.

El subgrafo predecesor de una Búsqueda Primero en Profundidad forma un primer bosque de profundidad compuesto por diversos árboles primero en profundidad. Los arcos en E_π son llamados arcos de árboles.

Como en la Búsqueda Primero en Anchura, los vértices son coloreados durante la búsqueda para indicar su estado. Cada vértice es inicialmente blanco, es coloreado de gris cuando es descubierto y de negro cuando es finalizado, esto es cuando la lista de adyacencias es examinada completamente. Esta técnica garantiza que cada vértice termina hacia arriba en un primer árbol de profundidad, de tal modo que estos árboles sean disyuntos.

Al lado de la creación de un primer bosque de profundidad, el método también timestamps cada vértice. Cada vértice *v*, tiene dos timestamps: el primer timestamp d[v] graba cuando *v* es descubierto, y el segundo timestamp f[v] graba la búsqueda finaliza. Estos timestamp son utilizados en varios algoritmos de grafos y son ventajosos al razonar sobre el comportamiento de la Búsqueda Primero en Profundidad.

El pseudocódigo del algoritmo es el siguiente:

DFS(G)[5]

1 **For** each vertex u ∈ V[G]
2 **Do** color[u] ← white
3 π[u] ← nil

[5] Ibid 4

```
4 time ← 0
5 For each vertex u ∈ V[G]
6     Do if color[u] = white
7       Then  DFS_visit(u)
```

DFS_visit(u)

```
1 color[v] ← gray
2   d[u] ← time + 1
3       For each v ∈ Adj[u]
4           Do if   color[v] = white
5               Then π[v] ← u
6                       DFS_visit(v)
7                        color[u] ← black
8                       f[u] ←  time  ← time + 1
```

El método DFS opera de la siguiente manera:

Las instrucciones 1 – 3 pinta todos los vértices de blanco e inicializa los π campos a nulo. La instrucción 4 resetea el contador de tiempo global. Las instrucciones 5 – 7 verifican cada vértice en V alternadamente, cuando un vértice blanco es encontrado, visita éste, usando DFS_visit. Cada vez que DFS_visit(u) es llamado en la instrucción 7, el vértice u inicia como raíz de un nuevo árbol en el primer bosque de profundidad. Cuando DFS retorna, cada vértice *u* ha sido asignado a un tiempo de descubierto d[u] y a un tiempo de finalización f[u].

En cada llamado DFS_visit(u), el vértice *u* esta inicialmente en blanco. La instrucción 1 pinta *u* de gris, y la 2 graba el tiempo de descubierto d[u] incrementando y almacenando la variable global time. Las instrucciones 3 – 6 examinan cada vértice *v* adyacente a *u* y recursivamente visita *v* si este es blanco. Como cada vértice v ∈ Adj[u] es considerado en la instrucción 3, se dice que el arco (u, v) es explorado por la Búsqueda Primero en

Profundidad. Finalmente, después de que cada arco que sale de *u* ha sido explorado, las instrucciones 7 – 8 pintan a *u* de negro y graban el tiempo de finalización f[u].

2.4.3 Algoritmos Evolutivos. Los algoritmos evolutivos están inspirados en la Teoría Evolutiva propuesta por Charles Darwin, en la cual se explica el origen y evolución de las especies, en esta teoría, Darwin introduce tres componentes principales de la evolución: replicación, variación y selección natural.

Un *Algoritmo Evolutivo* es un método de optimización estocástico, en el cual un grupo de estructuras de datos, son sometidas a una serie de operadores que simulan el proceso evolutivo, propuesto por Charles Darwin. Las estructuras de datos (individuos), representan las soluciones del problema que se está resolviendo. En cada ciclo (generación) del algoritmo se seleccionan las mejores soluciones. Estas soluciones, se reproducen entre sí, combinando sus características y generando así nuevas soluciones que serán evaluadas en la siguiente generación del Algoritmo[6].

Básicamente un algoritmo evolutivo presenta el siguiente esquema:

Inicio

```
 generar_población_inicial()
 mientras que(no se cumpla criterio de paro) haga
    calcular_fitness()
    selección()
    crossover()
    mutación()
fin mq
fin proc
```

La variante más conocida de los algoritmos evolutivos son los Algoritmos Genéticos. En los Algoritmos Genéticos, una solución es codificada como una cadena de 0s y 1s de

[6] ALVARADO, Carolina y HERAZO Ivan. Análisis comparativo de los algoritmos evolutivos NSGA-II Y SPEA-II mediante la solución de un problema de optimización multiobjetivo.190 p. Trabajo de Grado (Ingeniero de Sistemas). Universidad del Norte, Facultad de Ingeniería de Sistemas. Barranquilla, Colombia 2004.

longitud fija. A esta cadena de 0s y 1s se le denomina *cromosoma* o *individuo*. Esta codificación es conocida como *genotipo*. La imagen de una solución a través de **f** se le conoce como *fenotipo*. Inicialmente en un algoritmo genético existe una población inicial de soluciones (cromosomas) generados aleatoriamente y que se espera que a través de dos operadores llamados *cruce* y *mutación*, esta población mejore y se encuentren mejores soluciones. He aquí la analogía con la Teoría de Evolución Natural. Los individuos mejor adaptados son los que sobreviven y tienden a reproducirse. Los menos adaptados desaparecen.

La manera en que trabajan los Algoritmos Genéticos pueden ser explicados a través de dos conceptos: *Esquema* y *Bloques Constructivos*.

Un *Esquema* es un hiperplano en el espacio de búsqueda, que permite la exploración de las similitudes entre los cromosomas. La representación común de ellos usa además de 0's y 1's, un tercer símbolo, "*", que significa el símbolo sin importancia (*don't care symbol*). Un esquema representa entonces todas las cadenas que cazan en las mismas posiciones donde se encuentran los 0's y los 1' del esquema. Por ejemplo, se tiene un problema que codifica las soluciones usando 4 bits y se tiene el esquema (1,*,*,0). Los individuos (1,0,0,0) , (1,0,1,0) , (1,1,0,0) , (1,1,1,0) cazarían con el esquema anterior. De aquí se nota que un esquema caza exactamente con 2^r donde r es el número de símbolos sin importancia.

Holland mostró que el comportamiento de los algoritmos genéticos era más fácil describirlos en términos de esquemas. Holland fue capaz de demostrar que una población de tamaño n procesaría usualmente $O(n^3)$ esquemas (complejidad en espacio). Este resultado, conocido como Paralelismo Implícito, es ampliamente reconocido como uno de los factores principales en el éxito de los algoritmos genéticos [7].

[7] SMITH, James Edward. Self Adaption in Evolutionary Algorithms. 112p.Trabajo de Grado (Doctor of Philosophy). Faculty of Computer Studies and Mathematics. University of the West England. Bristol, England, 1998.

De esta forma se formuló un teorema que se conoce con el nombre de Teorema de los Esquemas (Schema Theorem), que dice que esquemas cortos, de bajo orden (longitud corta) reciben crecimiento exponencial en generaciones subsecuentes de un algoritmo genético.

Junto con lo anterior, se formuló entonces la hipótesis de los bloques constructivos, la cual dice que un algoritmo evolutivo busca cerca de los óptimos por la yuxtaposición de los esquemas cortos, de bajo orden y amplio performance, llamados Bloques Constructivos[8].

Los algoritmos genéticos funcionan fortaleciendo y encontrando bloques constructivos. A partir de ellos surgen las soluciones óptimas del problema de optimización que se está resolviendo. Krasnogor propone que en las búsquedas locales los algoritmos meméticos (en especial los algoritmos meméticos Auto Generados) son los generadores de los bloques[9].

- **Elementos de un Algoritmo Evolutivo.** La diferencia entre un algoritmo genético y un algoritmo evolutivo, radica en que la representación de las soluciones en un algoritmo evolutivo no es tan estricta como en un algoritmo genético, es decir, en un algoritmo evolutivo se pueden utilizar representaciones más naturales y apropiadas al tipo de problema que se está tratando de resolver.

A continuación se describe cada uno de los componentes que intervienen en un algoritmo evolutivo:

- **Individuo:** Es la representación de una solución del espacio de soluciones factibles de un problema. Un individuo es una estructura de datos tal como una cadena de bits, un árbol, un grafo, etc.

[8] MICHALEWICZ, Zbigniew. Genetic Algorithms + Data Structures = Evolution Programs. Editorial Springer-Verlag. New York, 1994. p.53.
[9] KRASNOGOR, Natalio y GUSTAFSON, Steven. La Búsqueda Local como Proveedora de Bloques Constructivos en Algoritmos Meméticos Auto-Generados. Automated Schedulling, Optimization and Planning Group. School of Computer Science and IT. University of Nottinngham, United Kindom. 2003.

- **Población:** La población en un algoritmo evolutivo es un conjunto de individuos o soluciones. Se espera que a través la aplicación de los operadores de Cruzamiento y Mutación en un Algoritmo Evolutivo se encuentren las soluciones óptimas del problema de optimización. La población inicial es generada aleatoriamente.

- **Función de Aptitud o Fitness:** Es un valor dado a un individuo que mide qué tan bien éste esta adaptado al entorno o problema, en otras palabras, el fitness mide qué tan bueno es un individuo dentro de la población en la que se encuentre y por tanto, determina la posibilidad de ser seleccionado para cruzamiento.

- **Operadores Genéticos.** Estos operadores son los que simulan el comportamiento evolutivo de las especies y se aplican sobre los individuos de la población con el fin de mejorarla. Los operadores más comunes son:

- **Selección:** En esta etapa se escogen o seleccionan los individuos más aptos para el cruzamiento y la mutación, con el fin de generar una nueva población. Existen varios métodos para realizar la selección, a continuación explicaremos algunos de ellos:

- ***Método de la Ruleta:***

  Sea $\boldsymbol{\Theta}$ el conjunto de soluciones factibles (en nuestro caso individuos), y **F**: $\boldsymbol{\Theta} \rightarrow \mathbf{R}$ la función de aptitud o fitness, entonces la probabilidad de que un individuo $x_i \in \boldsymbol{\Theta}$ sea seleccionado es igual a:

$$P(x_i) = \frac{F(x_i)}{\sum_{j=1}^{n} F(x_j)} \qquad \textbf{(18)}$$

  Donde *n* es el tamaño de la población, *x* un individuo y F(xj) el fitness del j-ésimo individuo de la población.

Luego de haber calculado para cada individuo de la población, esta probabilidad, se genera en forma aleatoria un número y $\in$ [0, 1]. Se suman los $P(x_i)$ obtenidos hasta que se rebase el número y, el elemento que rebasó el número generado, es escogido.

- ***Método del Torneo***:

  Sea $\boldsymbol{\Theta}$ el conjunto de soluciones factibles (en nuestro caso individuos), y **F**: $\boldsymbol{\Theta} \rightarrow \mathbf{R}$ la función de aptitud o fitness,. Dado el conjunto $\boldsymbol{\psi} \subset \boldsymbol{\Theta}$, entonces se selecciona el elemento $x \in \boldsymbol{\psi}$ que satisfaga F(x) = max(F(y) para todo y $\in \boldsymbol{\psi}$**.** Es decir, se seleccionan *j* elementos de la población aleatoriamente conformando con ello un subconjunto y luego se escoge de allí, aquel elemento que tenga mejor fitness. Una vez realizada la escogencia, elementos no escogidos son devueltos a la población, para que puedan ser seleccionados una vez más. Cuando *j* = 2, se denomina *Torneo Binario*.

- **Cruzamiento:** El Cruzamiento es un operador en el que dado dos individuos de la población (padres), se genera uno o más individuos (hijos) con la particularidad que ellos "heredan" características de ambos padres.

- **Mutación:** La Mutación es un operador en el que dado un individuo, se modifica algo de él para crear otro individuo, este operador es necesario para mantener la heterogeneidad de la población.

- **Elitismo:** El elitismo consiste en tomar a los mejores individuos de cada población y agregarlos directamente a la siguiente población sin necesidad de que pasen necesariamente por el cruzamiento. El elitismo es opcional en un algoritmo evolutivo.

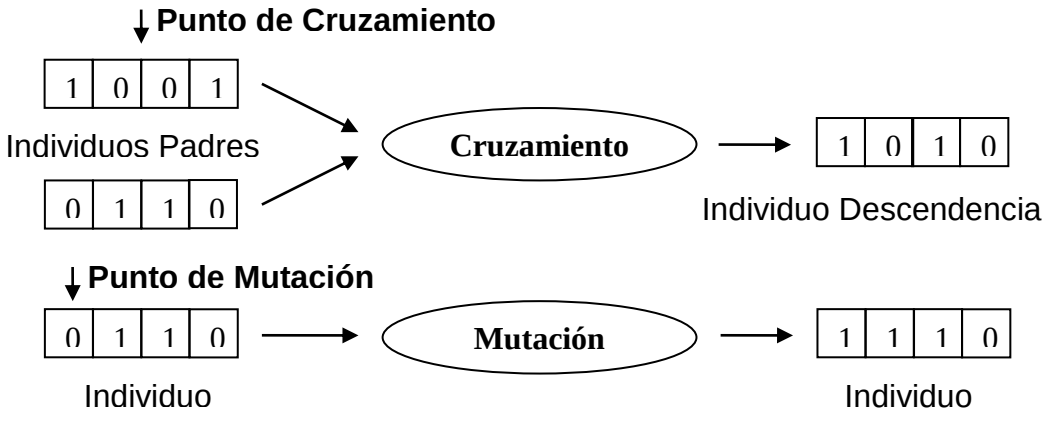


Figura 2 Operadores de Cruzamiento y de Mutación aplicados a individuos

- **Algoritmos Evolutivos en Optimización Multiobjetivo.** El primer algoritmo desarrollado para optimización multiobjetivo fue propuesto por J. David Schaffer en 1984, se llamó VEGA (Vector Evaluated Genetic Algorithm). VEGA es un algoritmo que fracciona la población en P/k subpoblaciones, donde P, es la población total y *k*, es el número de funciones objetivos, asignando como fitness a un individuo de la subpoblación j– ésima, el valor que tenga la función objetivo j – ésima. Luego se mezclan todos los individuos sobre los que se aplican crossover y mutación.

Es decir, el algoritmo no tiene en cuenta la definición de Optimización Pareto directamente dentro de él, ya que evalúa cada objetivo independientemente, esto trae como desventaja que las soluciones encontradas por él se concentran hacia los extremos del Frente de Pareto, obteniendo una no muy buena distribución del mismo, es decir, no mantiene la diversidad en él, porque al evaluar una sola dimensión de los objetivos, elementos no dominados pueden ser eliminados.

Cuando sólo se considera un objetivo aislado, en el que cada vez, una solución es evaluada sobre dicho objetivo para la selección, se dice que se ha realizado una ***Selección por Criterios****, como en el caso de* VEGA.

Un algoritmo evolutivo en optimización multiobjetivo no sólo debe encontrar soluciones del Frente Pareto, sino que también debe mantener una buena distribución del mismo, con el fin de tener diversidad en el Frente Pareto, así, a partir de esta necesidad, han surgido distintas clases de algoritmos evolutivos multiobjetivos basados en la dominancia de Pareto, en donde:

- Se utiliza el Orden Parcial de la definición de Optimización de Pareto directamente para realizar la selección. Esto se conoce como ***Selección Pareto***. NSGA, SPEA, (1+1) PAES, M - PAES son ejemplos de esta clase de algoritmos.

- Se convierte en un número escalar las diferentes funciones objetivos, usando una función paramétrica, y variar dichos parámetros de tal forma, que un conjunto diverso de soluciones pueda ser encontrado. Esta clase se conoce como ***Selección Escalar***. Ejemplos de esta clase de algoritmos son adaptaciones del Recocido Simulado y de la Búsqueda Tabú en optimización multiobjetivo al igual que el MOGLS y el RD - MOGLS.

Debido a que en este trabajo se desarrollará un algoritmo híbrido en que la selección es la que realiza el algoritmo SPEA 2, explicaremos a continuación dicho algoritmo evolutivo. El SPEA 2 es un algoritmo evolutivo que incorpora una población elitista de tamaño fijo en todas las generaciones.

- **Strength Pareto Evolutionary Algorithm 2 (SPEA 2).** El SPEA 2 es un algoritmo evolutivo utilizado para Optimización Multiobjetivo creado por Zitzler,

Laumanns y Thiele[10] en el año 2001. Este algoritmo mejora al algoritmo SPEA propuesto por Zitzler, Laumanns en 1999.

SPEA 2 mantiene una población elitista en la que guarda los mejores elementos de la población en cada generación. Los elementos de la población elitista puedenj estar conformados por elementos no dominados exclusivamente, o por la combinación de elementos no dominados y los mejores elementos dominados, esto se debe a que en el SPEA 2 la población elitista es de tamaño fijo.

A continuación, la figura 3 muestra el ciclo principal del algoritmo SPEA y la figura 6 del SPEA2.

Entrada:	N	*(Tamaño de la Población)*
	$\overline{N}$	*(Tamaño del archivo)*
	T	*(Número máximo de Generaciones)*
Salida:	***A***	*(Conjunto no dominado)*
Paso 1:	***Inicialización:*** *Generar la población inicial* P_0 *y crear el archivo vacío (Conjunto Externo)* $\overline{P}_0 = \phi$. *Colocar t=0*	
Paso 2:	***Asignación de Fitness:*** *Calcular los valores de fitness de los individuos en* P_t *y* $\overline{P}_t$	
Paso 3:	***Selección Ambiental:*** *Busca las soluciones no dominadas en* $P_t \cup \overline{P}_t$ *y agregar las soluciones no dominadas de* $P_t \cup \overline{P}_t$ *a* $\overline{P}_{t+1}$. *Si el tamaño de* $\overline{P}_{t+1}$ *excede a* $\overline{N}$, *reducir* $\overline{P}_{t+1}$ *mediante clustering.*	
Paso 4:	***Terminación:*** *Si* $t \geq T$ *o se satisface otro criterio de paro entonces retorne el conjunto A de soluciones no dominadas tomadas de* $\overline{P}_{t+1}$.	
Paso 5:	***Selección:*** *Realizar torneo binario con reemplazo en* $\overline{P}_{t+1}$ *para llenar el conjunto de apareamiento.*	
Paso 6:	***Cruzamiento:*** *Aplicar los operadores de recombinación y cruzamiento al conjunto de apareamiento y construir* P_{t+1}. *Incrementar t=t+1 e ir al paso 2.*	

Figura 3 Ciclo Principal de SPEA [11]

[10] ZITZLER, Eckart; LAUMANNS, Marco and THIELE Lothar. SPEA2: Improving the Strength Pareto Evolutionary Algorithm. Zurich: Swiss Federal Institute of Technology (ETH), 2000. Computer Engineering and Networks Laboratory (TIK) Report No. 103.
[11] Ibid 10.

Las principales diferencias entre el SPEA y el SPEA 2 son las siguientes:

- Un esquema mejorado de asignación de Fitness que toma en cuenta a cuántos individuos domina un individuo y por cuántos individuos es dominado ese individuo.

- Una técnica de estimación de densidad del vecino más cercano.

- Un nuevo método de truncamiento en la población elitista que garantiza la preservación de soluciones en el Frente Pareto.

El SPEA 2 tiene los siguientes pasos:

- **Inicialización:** La inicialización de los individuos del SPEA 2 es de manera aleatoria.

- **Asignación de Fitness:** El fitness de un individuo en el SPEA 2 es igual a

$$F(\boldsymbol{i}) = R(\boldsymbol{i}) + D(\boldsymbol{i}) \qquad \textbf{(19)}$$

donde *R(i)* es una función llamada *Raw Fitness* (Fitness Bruto) y *D(i)* es una función de estimación de Densidad al vecino *k* –ésimo vecino mas cercano.

Entrada:	N	*(Tamaño de la Población)*
	$\overline{N}$	*(Tamaño del archivo)*
	T	*(Número máximo de Generaciones)*
Salida:	***A***	*(Conjunto no dominado)*

Paso 1: ***Inicialización:*** *Generar la población inicial* P_0 *y crear el archivo vacío (Conjunto Externo)* $\overline{P}_0 = \phi$ *. Colocar t=0*

Paso 2: ***Asignación de Fitness:*** *Calcular los valores de fitness de los individuos en* P_t *y* $\overline{P}_t$

Paso 3: ***Selección Ambiental:*** *Copiar todos los individuos no dominados de* P_t *y* $\overline{P}_t$ *a* $\overline{P}_{t+1}$ *. Si el tamaño de* $\overline{P}_{t+1}$ *excede a* $\overline{N}$ *, reducir* $\overline{P}_{t+1}$ *mediante el operador de truncamiento, de lo contrario, si el tamaño de* $\overline{P}_{t+1} <$ $\overline{N}$ *, entonces llenar* $\overline{P}_{t+1}$ *con individuos dominados de* P_t *y* $\overline{P}_t$ *.*

Paso 4: ***Terminación:*** *Si* $t \geq T$ *o se satisface otro criterio de paro entonces retorne el conjunto A de soluciones no dominadas tomadas de* $\overline{P}_{t+1}$ *.*

Paso 5: ***Selección:*** *Realizar torneo binario con reemplazo en* $\overline{P}_{t+1}$ *para llenar el conjunto de apareamiento.*

Paso 6: ***Cruzamiento:*** *Aplicar los operadores de recombinación y cruzamiento al conjunto de apareamiento y construir* P_{t+1}*. Incrementar t=t+1 e ir al paso 2.*

Figura 4 Ciclo Principal de SPEA 2 [12]

R(i) se calcula de la siguiente manera:

$$R(i) = \sum_{j \in Pt \,(\, Pt,\, i \prec j} S(i) \qquad \textbf{(20)}$$

donde *S(i)* se define como

$$S(i) = |\{ j \in P \cup \overline{P} \mid i \prec j \mid \qquad \textbf{(21)}$$

| . | denota la cardinalidad del conjunto y a *S(i)* se le denomina *Strength.*

D(i) es una función que adapta el método del *k*-ésimo vecino mas cercano, donde la densidad en un punto es una función (decreciente) de la distancia al *k*-ésimo vecino

[12] Ibid 11

mas cercano. En el SPEA 2 se toma el inverso de la distancia como la estimación de densidad.

De esta manera, *D(i)* es igual a :

$$D(i) = \frac{1}{\sigma_i^k + 2} \qquad \textbf{(22)}$$

σ_i^k es la distancia al k-ésimo vecino mas cercano y $k = \sqrt{N + \overline{N}}$. *N* es el tamaño de la población *P* y $\overline{N}$ el tamaño de la población elitista.

- **Selección Ambiental:** En la etapa de selección ambiental, se copian primero todos los elementos no dominados. Éstos son aquellos cuyo fitness es menor que 1, es decir, $\overline{P}_{t+1} = \{i \mid i \in P_t \cup \overline{P}_{t+1} \wedge F(i) < 1\}$. Si el tamaño del frente no dominado es igual a $\overline{N}$, el procedimiento de selección ambiental termina. De lo contrario, pueden ocurrir dos posibilidades: Si el tamaño es menor que $\overline{N}$, se copian los mejores individuos de la población P_t, es decir, aquellos individuos de P_t cuyo fitness sea más bajo hasta que el tamaño de la población elitista sea igual a $\overline{N}$. Si el tamaño de la población elitista es mayor que $\overline{N}$, entonces se invoca el procedimiento de truncamiento el cual remueve iterativamente individuos de $\overline{P}_{t+1}$ hasta que el tamaño de $|\overline{P}_{t+1}| = \overline{N}$. En cada iteración del procedimiento de truncamiento, un individuo *i* es escogido para ser removido si cumple con $i \leq_d j$, para todos los $j \in \overline{P}_{t+1}$

$$i \leq_d j \Leftrightarrow \forall 0 < k < |\overline{P}_{t+1}| : \sigma_i^k = \sigma_j^k \vee \exists 0 < k < |\overline{P}_{t+1}| : \left[\forall 0 < l < k : \sigma_i^k = \sigma_l^k\right] \qquad \textbf{(23)}$$

donde σ_i^k denota la distancia de *i* a su *k*-ésimo vecino mas cercano. En otras palabras, el individuo que tiene la distancia mínima a otro individuo es escogido en cada etapa. Si hay varios con la distancia mínima, el empate se rompe considerando las segundas menores distancias y así se continúa.

- **Terminación:** Si se cumple con el criterio de paro del algoritmo, se devuelve como respuesta la población elitista $\overline{P}_{t+1}$.

- **Selección de Elementos para Cruzamiento:** La selección de los elementos para cruzamiento se realiza mediante torneo binario con reemplazo sobre la población elitista $\overline{P}_{t+1}$, es decir, todos los padres se escogen de la población elitista. Se realiza esta selección hasta que se llene el conjunto de elementos para cruzamiento.

- **Variación:** Se aplican los operadores de cruzamiento y mutación para generar la población P_{t+1} siguiente.

2.5 OPTIMIZACION COLONIAS DE HORMIGAS (ACO)

Una heurística natural conocida como Sistemas de Hormigas (Ant Systems (AS)), fue introducida por Coloni, Dorigo y Maniezzo en 1991 - 1992, la idea básica de esta heurística es la de simular el comportamiento de un conjunto de agentes que cooperan para resolver un problema de optimización combinatoria a través de comunicación simple.

Las hormigas pertenecientes a las especies I. Humilis y Lasius Niger tienen la capacidad de encontrar un camino que los conduce de su colonia a la fuente de alimento y regresar a ésta, incluso pueden saltar obstáculos con el fin de lograr el sustento, todo ello con relativa facilidad. Esta capacidad es el resultado de emplear "comunicación química" entre las hormigas, esto se logra a través de una sustancia llamada feromona y el rastro de feromona dejado por la presencia de muchas hormigas cuando toman un camino que las conduce desde su colonia a una fuente de alimento, es decir, este rastro les indica la bondad del camino a tomar.

ACO, es un paradigma que permite diseñar algoritmos metaheurístico para resolver problemas de optimización combinatoria. La base de los algoritmos ACO consiste en la combinación de una información a priori acerca de una solución promisoria y la

información de la estructura de buenas soluciones obtenidas previamente. ACO es una clase de algoritmo cuyo primer componente es Sistemas de Hormigas (Ant Systems), inspirado en el comportamiento real de las hormigas y la búsqueda paralela sobre diversos hilos constructivos computacionales basados sobre datos de un problema local y una estructura de memoria dinámica que contiene información sobre la calidad de los resultados obtenidos previamente.

2.5.1 Optimización Combinatoria. Un problema de optimización combinatoria es un problema definido sobre un conjunto finito **C** = c_1, c_2, . . ., c_n de componentes básicos dados. Un subconjunto **S** de componentes, que representa una solución del problema; un subconjunto **F** de soluciones factibles, $\mathbf{F} \subseteq \mathbf{2^C}$, luego **S** es una solución factible si y solo sí $\mathbf{S} \in \mathbf{F}$. Una función z esta definida sobre el dominio de solución $z : \mathbf{2^C} \rightarrow \mathbf{R}$, el objetivo es encontrar una solución de costo mínimo factible **S*** , i. e. , encontrar **S***: $\mathbf{S} \in \mathbf{F}$ y $z(\mathbf{S^*}) \leq z(\mathbf{S})$, para toda $\mathbf{S} \in \mathbf{F}$[13].

El funcionamiento de un algoritmo ACO puede ser considerado como un conjunto de agentes computacionales (la colonia de hormigas), que operan en forma concurrente y asíncrona, que se mueven o cambian de estados (estados del problema)), como solución parcial del problema a resolver. El cambio de estado se realiza por medio de la elección de una decisión local aleatoria fundamentada en dos parámetros: el rastro de feromona (trail) y la atracción (attractiveness). Al moverse o cambiar de estado una hormiga, contribuye en la construcción de una solución, en donde la hormiga al completar la solución o durante la fase de construcción evalúa la solución para modificar el valor del rastro de feromona sobre los componentes que utiliza o utilizó en la solución. La importancia de esta información (rastro de feromona) es que puede ser buscada o utilizada por futuras hormigas. Además del rastro de feromona y el atractivo, un algoritmo ACO, incluye dos mecanismos adicionales: la evaporación de rastro (trail evaporated) y opcionalmente daemon actions. Con la evaporación de rastro, el algoritmo disminuye el rastro de

[13] MANIEZZO, Vittorio, GAMBARDELLA, Luca María y DE LUIGGI, Fabio. Ant Colony Optimization. Paper 2004.

feromona, evitando con ello la acumulación ilimitada sobre algunos componentes. El Daemon Actions se utiliza para implementar acciones centralizadas que no pueden ser realizadas por una sola hormiga, tales como la invocación de un procedimiento de optimización local, o la actualización de información global que se puede utilizar para predisponer el proceso de búsqueda desde una perspectiva local[14].

Vittorio Maniezzo, Luca Maria Gambardella, Fabio de Luigi[15] definen más específicamente el funcionamiento del algoritmo ACO como: Una hormiga es un simple agente computacional que iterativamente construye una solución para la instancia a resolver. Soluciones parciales del problema son vistas como estados. En la base del algoritmo ACO reside un bucle (loop), en donde para cada iteración, cada hormiga se mueve (realiza un paso) de un estado **ι** a otro **ψ,** correspondiente a una solución parcial. Esto es, en cada paso **σ**, cada hormiga k computa un conjunto $A_k\sigma(\iota)$ de expansiones factibles para el estado actual y moverlas cada una con una probabilidad. La distribución de probabilidad es especificada como sigue: para la hormiga k, la probabilidad $p^k_{\iota\,\psi}$ de moverse del estado **ι** al estado **ψ** depende de la combinación de dos valores:

- El atractivo de moverse η_ι , el cual es computado a través de alguna heurística indicando el movimiento a priori deseable.

- El nivel de rastro de feromona $\tau_{\iota\,\psi}$, el cual indica que tan eficiente o bondadoso ha sido el movimiento en el pasado: entonces esto nos representa un indicador a posteriori de qué tan deseable es el movimiento.

El rastro de feromona es actualizado usualmente cuando todas las hormigas han completado una solución, incrementando o decrementando el nivel de rastro correspondiente a moverse hacia buenas o malas soluciones respectivamente.

En la figura 5 se muestra el pseudo-código del algoritmo ACO :

[14] Ibid 13
[15] Ibid 14

```
Procedure ACO Metaheuristic
    Schedule Activities
        Construct_Ants_Solutions
        Update_Pheromones
        Daemon Actions  % optional
    End Activities
End Procedure
```

Figura 5 Pseudo-código de la metaheurística ACO

Como el problema que se quiere resolver es un modelo estático, el procedimiento Daemon Actions se implementará como una búsqueda local de soluciones (Apply Local Search), a continuación en la figura 6, mostramos la versión estática de la metaheurística:

```
Procedure ACO Metaheuristic Static
  Set parameters, initialize pheromone trails
    While (termination condition not met) do
        Construct_Ants_Solutions
        Apply_Local_Search
        Update_Pheromones
     End
End Procedure
```

Figura 6 Pseudos-código de la metaheurística ACO con Daemon Actions

2.5.2 Optimización multiobjetivo utilizando colonia de hormigas. La metaheurística para diseñar algoritmos de optimización basados en colonias de hormigas no fue concebida para tratar problemas multiobjetivo, por lo tanto se tiene que proponer una modificación

que permita solucionar el modelo propuesto en este trabajo. La implementación de este algoritmo fue realizada tomando como base la referencia [16] :

Para resolver el modelo planteado en la figura 6, se utilizará el grafo de construcción $G_c = (C, L)$ que tiene componentes C=N (nodos la red) y L=E (enlaces entre los nodos). A cada enlace $l_{ij} \in L$ se le asocia un vector de feromona $\tau^f{}_{ijs} \in R^{11}$ donde $f \in F$ y $0 < s \leq 11$, esto quiere decir que para cada flujo se maneja un vector de feromonas independiente. Las soluciones candidatas S son conjuntos de árboles que representa a un flujo de $f \in F$. Cada uno de los árboles serán conjuntos de caminos que inician en una fuente sf y finalizan en destino $t_f \in T_f$.

- **Estructura del algoritmo.** La estructura general del algoritmo se muestra en la figura 7.

p: es un numero real en el intervalo (0,1), que indica el porcentaje de feromona que se va a evaporar en la implementación se utiliza el porcentaje de 0.3.

NumAnts: es un número natural, que indica cuantas soluciones se van a construir en cada iteración del algoritmo.

MaxIt: es el numero máximo de iteraciones que se van a realizar.

Network: aquí se tiene toda la información de la red, la topología, el número de lambdas que se utilizan, el ancho de banda de una lambda, la información de cada flujo: fuentes, destinos y ancho de banda de cada flujo; el ancho de banda de cada enlace, la potencia del láser en cada nodo y el factor de atenuación.

Procedure Ant-Colony multiobjective

read p,NumAnts, MaxIt, network

[16] Caro, Luis y Rosado, Pierre. Optimización Multiobjetivo en ransmisiones multicast sobre redes ópticas usando algoritmos evolutivos, meméticos y ant-colony, 203 p. Trabajo de Grado (Ingeniero de Sistemas). Universidad del Norte, Departamento de Ingeniería de Sistemas .Barranquilla , Colombia 2005.

```
initialize Pheromone Trails()
initialize Empty (parteto Front Solutions) // se inicializa la lista de elementos no
                                              dominados y las soluciones.
it=1
    While(it<=MaxIt) do
        ConstructAntSolutions;
        SelectParetoFront;
        UpdatePheromoneTrail;
        PheromoneEvaporation;
    it=it+1
    End-While
Show ParetoFont
End-Procedure
```

Figura 7 Estructura general del algoritmo

En las siguientes secciones se explica cual es el funcionamiento de cada uno des estos métodos.

- **Construct Ant Solutions.** En este procedimiento se construyen las soluciones del problema. Para construir una solución factible al modelo planteado, se necesita, inicialmente, construir un conjunto de árboles factibles, los cuales son un conjunto de caminos factibles que tomaría el flujo F desde el nodo fuente sf hacia los destinos $t_f \in T_f$ .este proceso se realiza mediante la función Construct Path, la cual utiliza y recibe la fuente sf, y un conjunto de destinos Tf.

sf: nodo fuente del flujo f.

Tf: nodos destinos del flujo f.

Nlast: es un arreglo donde se guardan temporalmente todos los nodos adyacentes, es decir todos los que se pueden visitar desde el último nodo del camino sin generar ciclos.

Path: es una lista enlazada donde se van agregando los caminos.

Get Adjacent Nodes (): esta es una función que recibe un camino y regresa Nlast. Lo anterior lo realiza basándose en la topología que modela la red por lo tanto no se genera solo caminos factibles.

Decision Rule Function (): esta recibe un camino Path y un conjunto de nodos adyacentes Nlast y regresa la función de probabilidad de visitar un nodo adyacente. Debido a que a cada objetivo se le asocia una matriz de feromona, esta función le asigna mayor probabilidad a aquellos enlaces que tengan mejores valores en la matriz de feromonas, es decir, se aplica una especie de dominancia de Pareto entre los enlaces. Los enlaces dominantes presentarán mejores valores de feromonas y por consiguiente, mayor probabilidad.

P: función de probabilidad de visitar un nodo adyacente, la cual es calculada por Decision Rule Fuction.

Random Number(): genera un numero aleatorio entre cero y uno.

```
Fuction Construct Path (sf, Tf)
        Path.add (sf); //se agrega el nodo fuente a lista
        current=sf;
        while( current ∉ Tf )do
                Nlast=GetAdjacentNodes(current)
                If(Nlast == NULL)then//
                        lastVisited = getLastVisitedNode()
                        lastcurrentNeighboorhood =  getCurrentNeighbors(lastVisited)
                        if(lastCurrentNeighboorhood !=null)
                                        current = lastVisited
```

```
                    else
                            Return NULL
                    end if
            Else
                    P = DecisionRuleFuction(Path,Nlast)
                    Path.addNode( Path )
                    If( P ∈ Tf)
                        Return P
                    End If
            EndIf
        End-While
        Return Path
End-Fuction
```

Figura 8 Estructura de construct Path

La construcción de una solución, es un proceso iterativo que construye árboles para cada flujo. Igualmente, los árboles se construyen iterativamente, a partir del nodo fuente y de los destinos de un flujo se construyen los caminos. Así, cuando se tiene una solución completa factible, se agrega en la lista de soluciones.

Solutions: es un arreglo de soluciones. Cada solución es una lista de árboles Af del flujo $f \in F$ y cada árbol es una lista de caminos.

```
Procedure ConstructAntsSolutions(sf, Tf)
        i=0
        For i=1 to NumAnts do
                For each f ∈ F do
                        Current Trees.clear()//inicializa la lista de caminos vacía.
                        While(Tf ≠ ϕ)do
```

```
                    currentPath= ConstructPath(sf, Tf);
                    if(currentPath==NULL)do
                            Path does not exist. Exit program.
                    else
                            Af.add(currentPath;)
                            Tf = Tf - {currentPath.getLastElement}; //aqui se
                            suprime de Tf  el destino del  camino creado.
                    end_if
                End-While
                currentTrees.add(Af)
            End-For
            Solutions.add(currentTrees);
        End-For
End-Fuction
```

Figura 9 Estructura de construct Ant Solutions

- **Select Pareto Front.** Es el método que en cada iteración *i*, a partir de los elementos en la lista de soluciones Solutions y del frente Pareto encontrado hasta el momento guardados en paretoFront, encuentra todos los elementos no dominados actualizando la lista paretoFront.

- **Update Pheromone Trail.** En este método se agrega una cantidad de feromona a cada enlace $l_{ij} \in L$ y el respectivo flujo $f \in F$. Para agregar la cantidad de feromona se utiliza el método de actualización del feromona del HC-ACO propuesto por Blum, Roli y Dorigo[17], en el cual se normaliza la cantidad de feromona que se va a agregar en cada enlace utilizado en una solución. Esto, es para todas las soluciones K en la lista solutions, para cada árbol $A_f \in K$, para cada camino $C \in A_f$, si i,j son nodos adyacentes en el

[17] BLUM, C., ROLI, A., and DORIGO, M. HC-ACO: The Hyper-Cube Framework for Ant Colony Optimization. IRIDIA-Universit´e Libre de Bruxelles.

camino C (es decir se utiliza el enlace $l_{ij} \in L$), entonces $\tau^{f}{}_{ijs} \leftarrow \tau^{f}{}_{ijs} + \frac{1}{\vec{f}_s(K)}$ $\forall s, 1 \leq s \leq 11$, donde . $\vec{f}_s(K)$ es la función objetivo numero s evaluada en las solución K.

- **Pheromone Evaporation.** Es un procedimiento que tiene como objetivo evadir la rápida convergencia del algoritmo en un frente Pareto subóptimo, aquí se reduce la cantidad de feromona p veces donde $p \in (0,1)$ y es un parámetro de entrada del algoritmo, matemáticamente esta reducción se expresa así para un enlace $l_{ij} \in L$: $\tau^{f}{}_{ij} \leftarrow p * \tau^{f}{}_{ij}$. La evaporación se realiza una vez por iteración sobre cada unos de los enlaces $l_{ij} \in L$, sin embargo $\tau^{f}{}_{ijs}$ para todo $1 \leq s \leq 11$, debe ser mayor que uno para que todos los enlaces tenga una probabilidad mayor que uno de ser escogidos. Por lo tanto solo se realiza evaporación sobre el s el vector de feromonas si no se viola esta condición.

2.6 REDES MULTICAST Y REDES ÓPTICAS

2.6.1 Generalidades sobre las Redes Multicast. Los diferentes tipos de comunicación que existen en una red de datos dependen del número de nodos transmisores y receptores que intervengan en la comunicación.

Las redes Multicast son redes que operan enviando información desde un nodo origen hacia múltiples destinos. Este tipo de redes son utilizadas en las transmisiones radiales en línea, transmisión de conferencia y muchas otras.

Para el envío de información en una red multicast, se han propuesto varias técnicas básicas sobre la dirección de enrutamiento de la información:

- Árboles de Steiner
- Enrutamiento Basado en el Origen
- Árboles con Puntos de Encuentro (Trees with rendezvous points).

La manera en que trabaja una transmisión multicast en el Internet es basada en alguno de los protocolos creados a partir de las técnicas anteriores y en el envío a *grupos multicast* (un conjunto de equipos y/o sitios) la información que se dirige a ellos. Los grupos multicast pueden ser de dos tipos:

- *Semipermanentes*: La única manera de que se modifique un grupo de este tipo es mediante una reconfiguración de la Red.

- *Dinámicos*: son aquellos grupos en los cuales se pueden hacer pequeños cambios sobre la marcha, una vez han sido configurados. Es decir, se pueden agregar o quitar equipos del grupo dinámicamente.

A continuación detallamos aspectos de las redes multicast:

- **Árboles de Steiner.** Para propósitos de enrutamiento, las redes de comunicaciones son modeladas como un grafo dirigido, donde los vértices representan los nodos de la red y los arcos (aristas) representan los enlaces entre dos nodos vecinos. Existen dos costos asociados a una red muticast, estos son los costos asociados a los destinos y el costo asociado por la utilización de la red en sí, que no es más que el costo por utilización de los enlaces (ancho de banda). Encontrar una solución óptima de enrutamiento multicast consiste en encontrar un árbol de enrutamiento donde la raíz es el nodo que envía la transmisión, los destinos pertenecen a dicho árbol y se busca que los costos asociados al árbol se minimicen.

Si el costo fuera únicamente el retardo de enviar un mensaje desde un origen a todos los destinos, el problema de enrutamiento multicast podría resolverse usando *k* veces (una vez por cada destino) el *Algoritmo de Dijkstra* lo que daría que el problema sería resuelto en

$O(kn^2)$ donde n es el número de nodos en la red. Sin embargo, cuando se habla de costo por utilización de los enlaces, el problema cambia; el problema se convierte en encontrar árboles de costo mínimo, que se conoce en la teoría de grafos como el *Problema de los Árboles de Steiner*[18].

Definición: Sea $G=(V,E)$ un grafo no dirigido, el arco $i,j \in E$ con costo real no negativo y sea N un conjunto de nodos, $N \subseteq V$. Un árbol T de G es llamado un **Árbol de Steiner** si contiene a todos los nodos de N.

El problema de los *Árboles de Steiner* consiste en encontrar un árbol de Steiner tal que su costo $c(T)=\Sigma_{ij \in E(T)} c(ij)$ es mínimo entre todos los árboles de Steiner. Los nodos que están en T pero que no están en N se les denominan *nodos de Steiner*. Cuando $| N | = 2$ se tiene el problema del camino más corto que puede ser resuelto usando el Algoritmo de Dijkstra en $O(n^2)$ y cuando $| N | = | V |$ se tiene el problema del árbol de recubrimiento mínimo que se puede resolver mediante el algoritmo de Kruskal o Prim también en tiempo polinomial $O(n^2)$. Para todos los demás casos, la complejidad del problema de los Árboles de Steiner es *NP-Duro*, aún cuando el costo de todos los enlaces es uno[19].

Utilizando este enfoque, el problema de enrutamiento multicast es un problema de árboles de Steiner con restricciones, tales como la raíz es el nodo transmisor y el grafo resultante de la red es un grafo dirigido. Aún así, el problema sigue siendo *NP-Hard*[20].

Existen algoritmos de aproximación que encuentran buenas soluciones al problema de los *árboles de Steiner*, sin embargo, debido a la complejidad del problema, y a que la solución del problema de los *Árboles de Steiner* optimiza de manera global, cualquier cambio en la

---

[18] Op. Cit. P 22

[19] CHERIYAN, J and RAVI, R. Approximation algorithms for network problems. 1998, 222 p. Disponible electrónicamente en http://www.math.uwaterloo.ca/~jcheriya/PS_files/ln-master.ps, p 84.

[20] GHANWANI, Anoop. Multipoint Routing and Traffic Management in High Speed Networks. 1998, 146 p Trabajo de Grado (Doctor of Philosophy). Department of Electrical and Computer Engineering. Duke University. California, USA, p. 44.

topología obliga a la recalcular una nueva solución óptima. Así, utilizar este enfoque lo hace poco atractivo para implementaciones prácticas de enrutamiento en redes de datos[21].

- **Enrutamiento basado en el origen.** El enrutamiento basado en el origen toma como premisa que los nodos receptores inician el cálculo de la información de enrutamiento. De esta forma, un árbol de recubrimiento se crea a partir de cada fuente (origen). El criterio de optimización en el árbol de enrutamiento es el Retardo Total o los Saltos Totales y el uso de los recursos no se toma en cuenta en el árbol. Dentro de esta técnica se pueden incluir los protocolos *DVMRP, MOSPF* y *PIM-DM.* Las características comunes de estos protocolos es que utilizan inundaciones periódicas, realizan podas para la construcción de árboles y que existe un árbol por cada origen.

  - **Distance Vector Multicast Roting Protocol (DVMRP).** Es el protocolo de enrutamiento multidifusión más antiguo. Este protocolo fue utilizado para realizar la implementación del **MBone**[22], siendo aún el predominante en él. Su construcción está basada en los principios del protocolo RIP para enrutamiento en unicast basado en el algoritmo de vector – distancia, extendiéndolo a multicast.

En su última versión, recurre a una técnica denominada multidifusión por trayectoria invertida (*RPM*, por sus siglas en inglés) para generar árboles de entrega para multidifusión IP. Cuando un enrutador recibe un datagrama en una interfase, el camino invertido al origen se chequea para saber si es la ruta más corta hasta ella utilizando una tabla de enrutamiento unicast de redes fuente que sean conocidas. Si el datagrama en cuestión, llegó desde el camino más corto desde la fuente, el enrutador lo difunde utilizando la técnica de inundación, sin enviarlo por el camino que lo devuelve a la fuente. Si no viene por el camino más corto, simplemente lo descarta, debido a que viene desde un camino redundante, nada óptimo.

[21] WITTMANN, R. and ZITTERBART M. Op. Cit. p 87.
[22] MBone. Abreviatura de Multicast Backbone. Es la red que soporta el envío de paquetes multicast en el Internet.

DVMRP incluye un protocolo unicast, utilizado para determinar la ruta más corta desde un enrutador cualquiera hasta la fuente, información que es intercambiada por todos los enrutadores en la red.

En la figura 10 se muestra el funcionamiento de *DRVMP*.

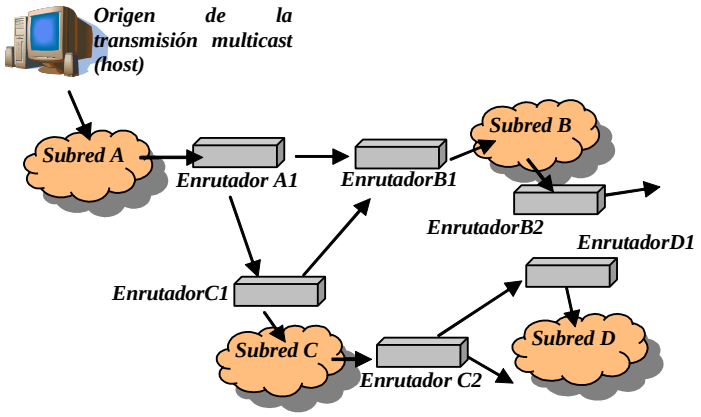


Figura 10 Enrutamiento por Vector Distancia

Fuente: MILLER, C. Kenneth. Multicast Networking and Applications.

El ejemplo de la figura 10, muestra el host de la subred A como el origen. Los datagramas enviados al enrutador B1 de la subred B llegan por dos interfases: directamente por la subred A (camino más corto) y a través del enrutador C1 de la subred C. El segundo de ellos es descartado, porque no llegó por la ruta más corta hasta la fuente. La misma lógica se aplica para las otras subredes.

Esta técnica permite a los datos multicast llegar a todas las subredes, posiblemente varias veces. Si un enrutador esta ligado a una subred que no desea recibir datos para un grupo multicast en particular, retorna un mensaje de poda a lo largo del árbol de distribución, lo cual impide el envío de datagramas subsecuentes como respuesta a máquinas que no sean miembros válidos de grupo. Igualmente, es posible añadir nuevos miembros al grupo multicast, insertando nuevas secciones al árbol mediante el uso de un mensaje de inserción.

Los mensajes de poda, tienen un tiempo de vida limitado. Por lo tanto, *DVMRP* re-inunda periódicamente, actualizando las rutas a un grupo cuando expira este tiempo. Cada pareja

origen – grupo y la interfase podada, requieren una entrada guardada en la memoria del enrutador para recordar que la interfase ya no hace parte del árbol (fue podada). Para evitar que esta información se guarde por siempre, se reconstruye el árbol re-inundando y repodando (si el origen aún está enviando datos al grupo relevante). Algunas características adicionales de *DVMRP* son las siguientes:

- Capacidad para realizar el proceso de túnel como parte del enrutamiento, la cual es indispensable para el funcionamiento de **MBone**. Las interfases se definen como físicas o por túnel en un enrutador que maneje este protocolo.

- Es un protocolo de enrutamiento multicast en modo denso.

- Actualmente, es soportado por muchos enrutadores comerciales.

o **Multicast Open Shortest Path First (*MOSPF*).** Este protocolo, provee extensiones multicast para el protocolo de enrutamiento unicast, *OSPF*, el cual es de pasarela interior (*IGP*) lo que quiere decir que, es utilizado para enrutar paquetes dentro de un sistema autónomo simple. *OSPF* está basado en el algoritmo de estado – enlace que permite cálculos rápidos de rutas con mínimo tráfico de protocolo de enrutamiento en la red.

Los enrutadores que funcionan con *OSPF* conocen todos los enlaces en la red, y usan esa información para calcular las rutas desde él mismo hasta todos los demás destinos dentro de ésta. *MOSPF*, funciona incluyendo información especial para multidifusión dentro de los anuncios de estado de enlace (*LSA*, por sus siglas en inglés) de *OSPF*, los cuales no son más que el mecanismo que este último utiliza para comunicar información de estado de enlace a los demás enrutadores. En *MOSPF*, se utilizan *LSA* multicast especiales, llamados de membresía de grupo, los cuales son usados para calcular rutas óptimas para el grupo

desde el origen[23]. Enrutadores designados en una subred son los encargados de comunicar la información de membresía de grupo a todos los demás, inundando la red con los *LSA* de membresía de grupo.

Cuando un datagrama multicast inicial llega, la subred origen está localizada en la base de datos de estado – enlace *MOSPF*, compuesta por su homóloga de *OSPF* más los *LSA* de membresía de grupo. La construcción de un árbol con las rutas más cortas cuya raíz está en el origen, es posible mediante cálculos hechos a partir de la información contenida en ella.

Los datos de los *LSA* de membresía de grupo, son utilizados para realizar la poda del árbol, asegurándose que las ramas que queden, lleguen únicamente a aquellas subredes que contengan miembros de grupo. El resultado de este proceso, es un árbol podado basado en la fuente cuya raíz es la subred emisora de paquetes.

MOSPF, también provee soporte para enrutamiento entre múltiples dominios *OSPF*, también llamados áreas, que pueden ser configuradas dentro de un sistema autónomo[24]. Esta estrategia, se implementa frecuentemente debido a que reduce los requerimientos computacionales en un enrutador individual y en consecuencia, mejora el rendimiento.

La figura 11, muestra el funcionamiento general de *MOSPF* por áreas *OSPF*.

[23] MILLER, C. Kenneth. Multicast Networking and Applications. United States: Addison Wesley, 1999. 282 p.
[24] Un sistema autónomo se define como: *"grupo de redes IP que poseen una política de rutas propia e independiente".*

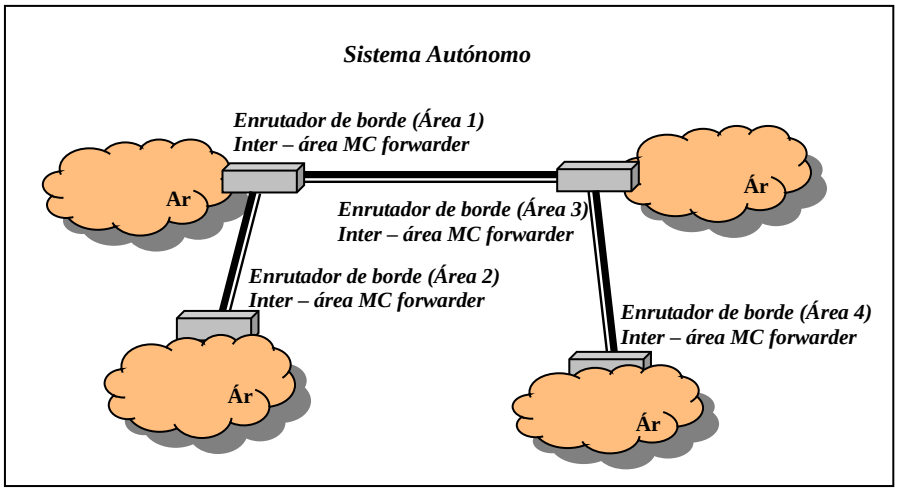


Figura 11 Funcionamiento *MOSPF* en áreas *OSPF*.

Fuente: MILLER, C. Kenneth. Multicast Networking and Applications.

MOSPF es una buena solución para multicast siempre y cuando los enrutadores disponibles en la red utilicen a su hermano para las transmisiones unicast (*OSPF*), dado que *MOSPF* es simplemente una extensión de este último. Si no es así, este protocolo no funcionará.

- **Protocol Independent Multicast – Dense Mode (*PIM-DM*).** Su modo de operar es similar al de *DVMRP*. Primero inunda, y luego empieza a podar hacia atrás las hojas del árbol donde no existan miembros de grupo. Según Miller[25], El funcionamiento de este protocolo, depende del protocolo de enrutamiento de unidifusión existente para determinar las rutas de regreso al origen, lo cual representa una gran diferencia con *DVMRP* que incluye su propio protocolo para realizar esta tarea.

Otra diferencia entre *DVMRP* y *PIM – DM*, es que los enrutadores *DVMRP* conocen a sus hijos (*downstream routers*), en el árbol multicast por cada grupo con raíz en el origen, y pueden determinar si esos hijos están en nodos hoja en el árbol. Utilizando esta información junto con una técnica conocida como recorte por horizonte dividido, es posible evitar

[25] Op. Cit. p 54

mensajes o enlaces duplicados, incluso antes de la poda. Por su parte, los enrutadores *PIM – DM* no guardan hijos e información de nodos hoja para todos sus enlaces, ahorrando recursos del enrutador, pero es posible que los paquetes multicast se repitan en el mismo enlace antes de que la poda tenga lugar.

- **Árboles con punto de encuentro (Trees with Rendezvous Points).** Los árboles con puntos de encuentro representan un nuevo desarrollo en el contexto de la comunicación multicast o multipunto. Los algoritmos que construyen estos árboles, están basados en el establecimiento de puntos de encuentro en la red. Estos puntos están familiarizados con la membresía de grupos y de esta forma ellos reciben información cuando un nuevo miembro se agrega al grupo. Cuando esta información se reenvía, pasa a través de todos los enrutadores localizados entre el nuevo miembro y el punto de encuentro.

De esta forma, un enrutador almacena información que provee detalles acerca de la membresía del grupo y los receptores. Si ningún miembro del grupo está localizado hacia un lado del enrutador, el enlace está inactivo. Esto está en contraposición con los algoritmos basados en el origen donde se hace inundación.

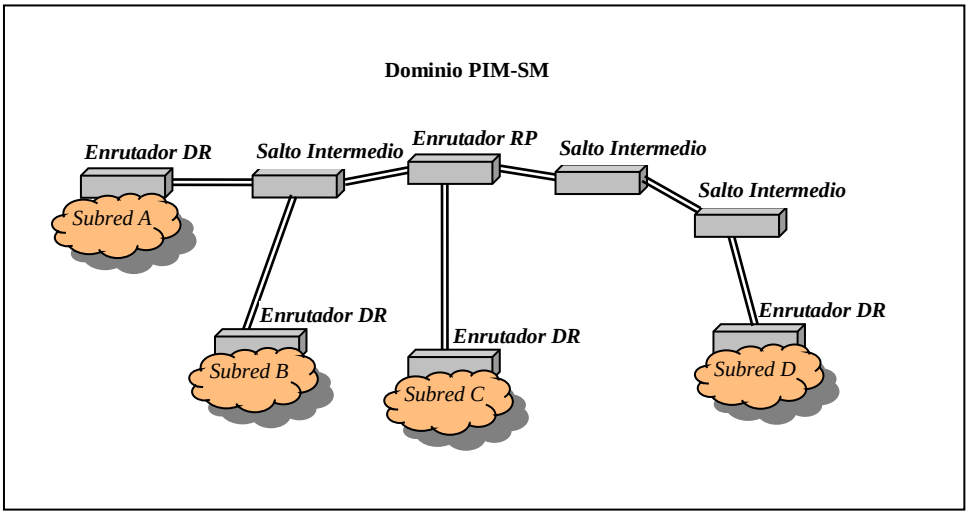
Comparados con los *árboles de Steiner*, sólo la selección de los puntos de encuentro óptimos representa un problema *NP-Completo*. El funcionamiento del flujo de información en esta técnica funciona de la siguiente manera: La información es enviada desde el origen hasta los puntos de encuentro y a partir de estos es distribuida hacia los receptores del grupo.

La ventaja de este método radica en la restricción de la distribución de la información a solamente los miembros del grupo. El esfuerzo involucrado en el mantenimiento de la información en los enrutadores es muy bajo. La desventaja de estos árboles es que concentran mucho tráfico en los puntos de encuentro y dichos puntos representan un punto de falla (es decir, si el punto de encuentro se deshabilita, no se podría enviar la información

desde el punto de encuentro hacía los miembros del grupo)[26]. Como protocolos basados en puntos de encuentro se pueden ver los siguientes: *PIM – Sparse Mode* y *CBT*.

- **Protocol Independent Multicast – Sparse Mode (*PIM-SM*)** . Derivado de *CBT*, fue diseñado junto con *PIM-DM*, para proveer un conjunto de protocolos de enrutamiento con formatos comunes de paquete, con el fin de dar soporte tanto para modo denso como para modo esparcido. En *PIM – SM*, los miembros de un grupo se unen en un punto de reunión (*RP*, por sus siglas en inglés). Los enrutadores con miembros adyacentes o los que siguen a estos últimos, son requeridos explícitamente para unirse a un árbol de entrega en modo esparcido transmitiendo mensajes de inclusión al *RP* asignado a ese grupo, como muestra la figura 12.

Figura 12 Dominio de PIM-SM



Fuente: MILLER, C. Kenneth. Multicast Networking and Applications.

Se selecciona un enrutador *PIM-SM*, que va ser el responsable del envío de los mensajes de poda y de unión al RP. El enrutador seleccionado es aquel que tiene la dirección IP más alta, y se conoce como enrutador designado (*DR*, por sus siglas en inglés).

[26] Op. Cit. p22

El enrutador designado, determina el *RP* al cual se asigna el grupo, y transmite un mensaje *PIM* de unión hacia al *RP* para el grupo. Enrutadores intermedios lo reenvían y crean una entrada de reenvío si no existe. En la figura 14, los nuevos miembros del grupo de las subredes A y B tienen mensajes reenviados desde la subred C directamente y a través del mismo enrutador intermedio y mensajes reenviados desde la subred D a través de 2 enrutadores intermedios diferentes.

- **CBT (Core – Based Tree)**. Permite mapear un grupo particular, a un núcleo particular de dos maneras diferentes. El primer método, utiliza un mecanismo conocido como **bootstrap**, El segundo, "implica la colocación manual de núcleos para crear árboles más óptimos; sin embargo esta técnica crea una significativa carga administrativa" [27].

A diferencia de *PIM – SM*, *CBT* no permite cambiar a un árbol de ruta más corta desde el árbol compartido, dado que sus desarrolladores creen que el mantenimiento de estado del enrutador en la red es un beneficio de escalabilidad importante que no debe ser comprometido.

- **MBone – MULTICAST BACKBONE.** MBone es la red que soporta el envío de mensajes multicast en Internet. Está basada en las direcciones IP clase D y en el protocolo IGMP como el protocolo de manejo de grupos.

La MBone fue inagurada en Marzo de 1992 a través de una transmisión multicast de audio de una sesión de la IETF (Internet Engineering Task Force). Desde entonces, más de 12000 subredes se han conectado a ella a en todo el mundo [28].

La MBone consiste en subredes multicast y enlaces entre dichas subredes, llamados *dominios* y *túneles*. En este contexto, una subred multicast es creada de puntos finales multicast y sistemas intermedios, llamados *islas.*

[27] Op. Cit. p54.
[28] Op. Cit. p 22.

Los sistemas multicast son sistemas que pueden trabajar con direcciones IP multicast, utilizar el protocolo IGMP y en el caso de los sistemas intermedios, llevan a cabo el enrutamiento multicast.

Los paquetes multicast pueden ser enviados dentro de dichos dominios. Existen, sin embargo, algunos dominios que están unidos al Internet a través de sistemas intermedios que no incorporan extensiones multicast. Así, no siempre es posible para los paquetes multicast el ser enviados y recibidos entre dominios. Este problema se resuelve enlazando los dominios al MBone a través de túneles.

Los túneles son conexiones virtuales entre un túnel multicast y los puntos finales. Enrutadores no multicast pueden ser localizados entre esos puntos finales. Los puntos finales son enrutadores multicast y se les denomina *mrouters*. Aquellos que no son multicast son llamados *unicast routers*. Un *mrouter* puede ser un enrutador "real" o una estación de trabajo/PC que ejecuta software de enrutamiento multicast. El MBone en sus principios trabajaba de la última forma y es por ello que su comportamiento era muy inestable, ya que los equipos no estaban optimizados para el paso de paquetes.

Debido al crecimiento del MBone, se ha vuelto un problema el mantenimiento del enrutamiento y la susceptibilidad a los fallos en las configuraciones. Así, la comunidad multicast se ha visto en la necesidad de desplegar configuraciones jerárquicas, como lo muestra la figura 13.

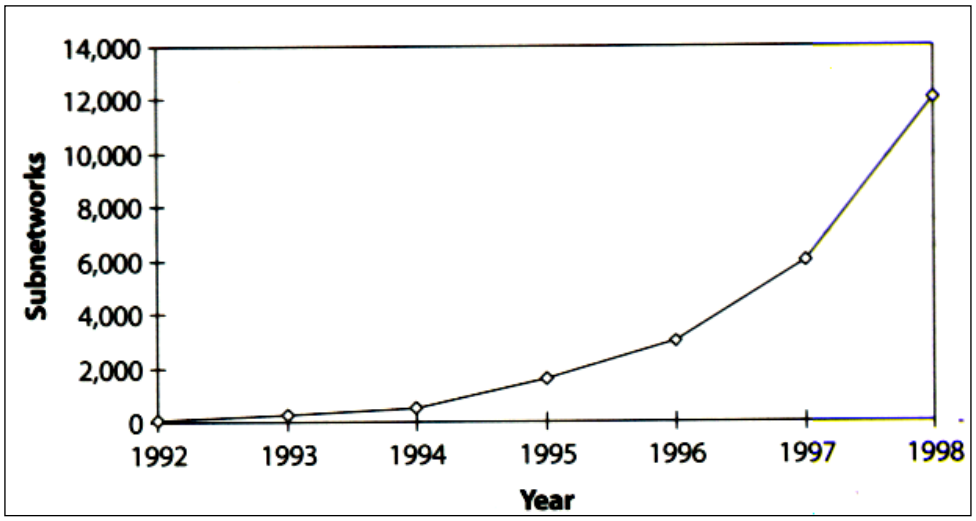


Figura 13 Crecimiento del MBone

Fuente: WITTMANN, R. and ZITTERBART M. Multicast Communication. Protocols and Applications. Muestra el crecimiento del **MBone.** Existen varios problemas con el MBone, a continuación se enunciarán varios de los problemas del MBone[29]:

- **Escalabilidad:** Con una red plana, los enrutadores de la red deben ser conocidos por cada enrutador. En su punto cumbre, MBone llegó a tener 10,000 enrutadores. Con este crecimiento, las tablas de enrutamiento se han vuelto inmanejables y el problema se ha manifestado en la inestabilidad del servicio MBone.

- **Manejabilidad:** Con el crecimiento del MBone, se ha vuelto difícil el manejo de la misma. No existe grupo formal para el manejo de la topología, y entonces se han evidenciado varias deficiencias, entre ellas:

    - ***Manejo con la Topología Virtual (Tunneling)*:** MBone se caracteriza por un conjunto de *islas* conectadas conjuntamente a través de túneles. El objetivo ha sido siempre el conectar estas islas juntas de la manera más eficiente posible, sin embargo, los túneles que se hacen son subóptimos. Así, dichos túneles son configurados de maneras ineficientes. Algunos

[29] ALMEROTH, Kevin. The Evolution of Multicast: From MBone to Inter-Domain Multicast to Internet2 deployment. Department of Computer Science, Unversity of California at Santa Bárbara. 1999. p. 11.

ingenieros, con es el caso de MCI, resolvieron el problema de los túneles reemplazándolos con un enlace sencillo físico.

- ***Políticas de Administración entre dominios:*** Debido a que el modelo de interconexión entre dominios es el establecer sistemas autónomos, el manejo de ellos representa un problema. Dichos sistemas son manejados o pertenecen a organizaciones distintas. Como resultado, entidades dentro de un sistema autónomo no son confiadas a otro sistema autónomo. Así, el intercambio de la información de enrutamiento entre dichos sistemas, se maneja cuidadosamente.

  El protocolo BGP (*Border Gateway Protocol)* es usado para intercambiar información entre sistemas autónomos, y así *BGP* se ha vuelto en un procedimiento aceptado cuando dos sistemas intentan comunicarse. El problema del manejo entre dominios consiste entonces en que los errores de enrutamiento se propagan a través de toda la red.

2.6.2 Generalidades sobre las redes ópticas. Los sistemas de transmisión óptica de alta velocidad comenzaron a desarrollarse con la invención del láser al transmitir ondas ópticas en espacio libre. Sin embargo, el primer sistema de transmisión de fibra óptica, no fue posible sino hasta los años 70, cuando se inventó el diodo semiconductor láser.

A medida que pasan los años, la velocidad de transmisión a través de fibra óptica, podríamos afirmar que se duplica por cada año que se avanza. Actualmente a pesar de que tenemos una alta capacidad de transmisión a través de fibra óptica, toda la capacidad de ésta no es utilizada, debido a que en el pasado la demanda de información no era tan alta como en la actualidad.

Con los últimos avances en la Ciencia de la Computación, la Electrónica y las Telecomunicaciones, servicios de información que son requeridos con una alta demanda tales como las video conferencias, la telemedicina, distribución de archivos de video,

imagen y texto, entre muchas más, implican el desarrollo de redes ópticas de áreas extensas, que permitan cumplir a cabalidad con los servicios de información actual requeridos.

Sí una red utiliza en sus enlaces de transmisión fibra óptica, no podría decirse que es una red óptica, porque la simple utilización de fibra óptica es apenas una condición necesaria, pero no suficiente para ser considerada como tal. Se llamará red óptica a una red de telecomunicaciones con enlaces de transmisión que son de fibra óptica, y además diseñada con una arquitectura que permita explotar las características únicas de las fibras.

Lo que se quiere realizar es una red que tenga un núcleo totalmente óptico, con sus enlaces conectados a través de nodos ópticos llamados ONN, además en el borde de dicho núcleo se tendrían estaciones de acceso a la red llamados NAS que serían la interfaz para que las terminales de los usuarios y otros sistemas no ópticos se conecten a la red óptica. Esta infraestructura se puede ver en la figura 14.

Los NAS suministran los puntos terminales (emisores y destinos) para la trayectoria de las señales ópticas dentro de la capa física. La comunicación afuera de la parte puramente óptica de la red, continúa de forma eléctrica ya sea terminando en un dispositivo final (por ejemplo las terminales del usuario) o atravesando equipos de conmutación electrónica (por ejemplo conmutadores ATM).

Los ONN son los que realizan las funciones de conmutación y enrutamiento que controlan la trayectoria de las señales ópticas, configurándolas para crear conexiones emisor-destino.

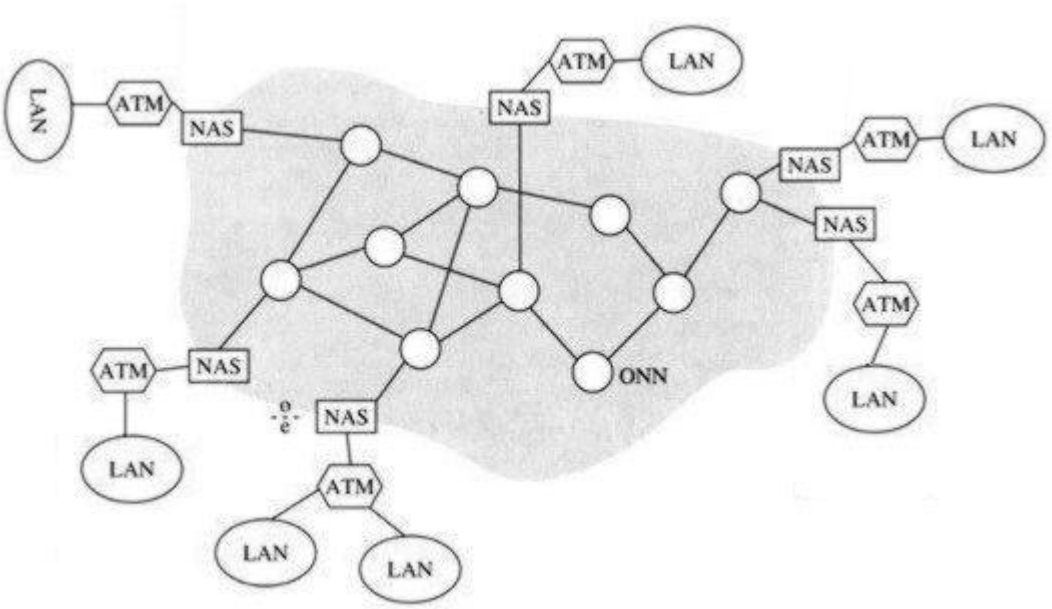


Figura 14 Infraestructura de una red óptica

Fuente: STERN, Thomas y BALA Krishna. Multiwavelength Optical Networks a Layered Approach, New Jersey, USA : Prentice Hall 2000.

2.6.3 Recursos de la red. El desempeño de una red está limitado por la cantidad y funcionalidad de sus recursos físicos. A continuación trataremos de explicar las diferentes funciones realizadas en una red óptica, enfatizando el papel de los recursos ópticos de proveer conectividad y flujo de información.

- **Capas y subcapas.** En la figura 15 podemos ver el ejemplo en capas de la red óptica planteada anteriormente.

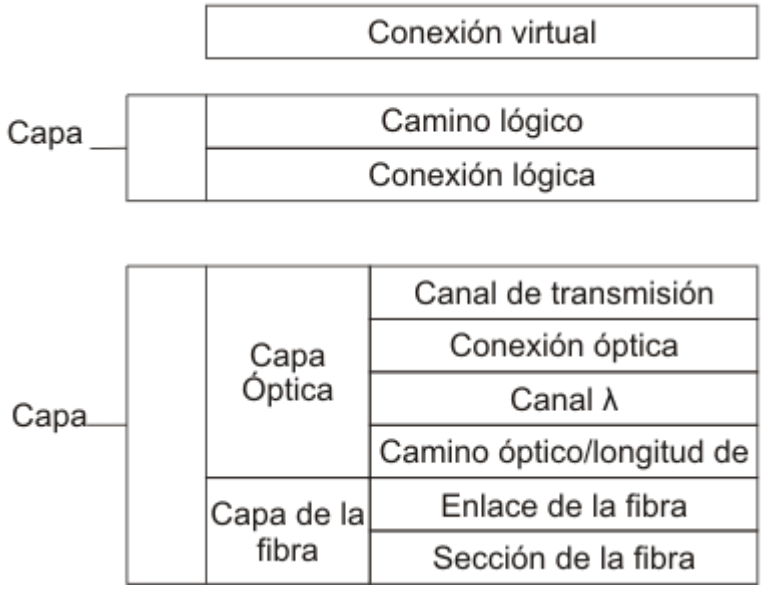


Figura 15 Modelo en capas de una red óptica

Fuente: STERN, Thomas y BALA Krishna. Multiwavelength Optical Networks a Layered Approach, New Jersey, USA : Prentice Hall 2000.

En términos de hardware, la interfaz entre la capa lógica y física se encuentra en los puertos externos (eléctricos) de los NASs. Una mirada detallada a una conexión típica punto a punto se presenta en la figura 15. Como se indicó, en las estaciones de acceso terminan las conexiones ópticas, y estas sirven de interfaz entre los equipos electrónicos y los ópticos.

En su lado óptico cada NAS se conecta a un ONN a través de un enlace de acceso, el cual consiste de uno o más pares de fibra. Las señales ópticas son intercambiadas entre la estación y la red a por transmisores-receptores en la estación. Cada nodo está conectado a sus nodos vecinos por una par de fibras, las cuales constituyen los enlaces internodales de la red. Un enlace muy largo puede tener uno o más amplificadores ópticos para compensar la atenuación de las fibras.

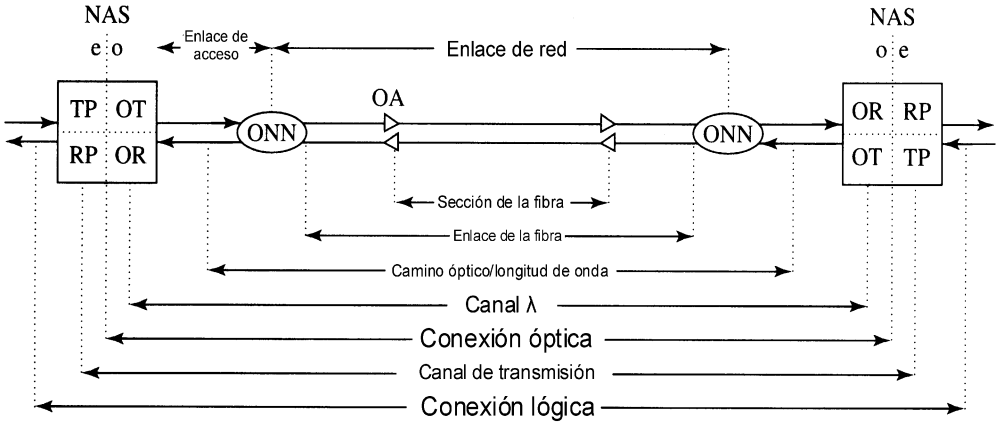


Figura 16 Modelo en capas de una red óptica

Fuente: STERN, Thomas y BALA Krishna. Multiwavelength Optical Networks a Layered Approach, New Jersey, USA : Prentice Hall 2000.

Las subcapas de la capa óptica funcionan de acuerdo a la forma en que el espectro óptico es particionado. La entidad más pequeña en la partición son los canales λ, a cada uno de los cuales se les asigna una longitud de onda distinta. Estos son los portadores de información básicos en la capa física. Se espera que los canales λ sean conmutados independientemente por los ONNs.

Cada conexión óptica punto a punto se realiza a través de un canal λ el cual es creado en dos pasos. El primero es asignar un canal λ al transmisor de la fuente y al receptor del destino en una longitud de onda seleccionada. El segundo paso es establecer un camino óptico (OP) a través de una secuencia de nodos que lleven esa longitud de onda de la fuente al destino.

Denominamos a la conexión unidireccional entre nodos externos de un par de NASs una conexión lógica (LC). Esta conexión lleva una señal lógica en un formato eléctrico

determinado. Todas las LC utilizan los recursos de la capa física, cada LC es transmitido a través de una conexión óptica por medio del canal de transmisión. El canal de transmisión realiza una conversión de la señal lógica a una señal de transmisión. Esta conversión toma lugar en el procesador de transmisión (TP), y la misma operación inversa se realiza en el procesador de recepción (RP). Además la señal de transmisión debe ser limitada al ancho de banda que pueda manejar el transmisor óptico (OT), el receptor óptico (OR), y el espectro permitido por el canal que lleva la señal. Es importante ver como se establecen las conexiones ópticas a través de las acciones coordinadas de los nodos y las estaciones.

La principal consideración al partir la capa óptica en subcapas es la operación de multiplexar, manejar múltiples accesos y conmutar. Usando la multiplexación varios canales lógicos se pueden combinar en un canal λ originado de una estación; usando múltiples LCs al mismo destino, y a través de conmutación, muchos caminos ópticos distintos pueden ser creados en diferentes fibras en la red, usando (y dejando de usar) canales λ de la misma longitud de onda. En la capa de la fibra el camino óptico es llevado por una sucesión de enlaces de fibra que conectan a los nodos de la res.

- **Multiplexación de la Red.** Un gran número de conexiones simultáneas se puede soportar en un enlace de una red óptica a través de la técnica de multiplexación que se use. Hay diferentes técnicas de multiplexación entre ellas están multiplexación por división de tiempo, espacio y frecuencia. En este trabajo supondremos que se usará DWDM que es multiplexación por división de longitud de onda densa, es una técnica de multiplexación como WDM, es decir por división de longitud de onda pero a diferencia de ésta, DWDM utiliza una separación más pequeña de las longitudes de onda.

Usando DWDM cada fibra tendrá diferentes conexiones en diferentes longitudes de ondas (canales λ), las longitudes de onda tienen que estar suficientemente separadas para evitar que se sobrepongan entre ellas lo cual produciría interferencia en la señal del receptor óptico, es por esto que en DWDM las longitudes de onda están separadas normalmente 1nm o 0.1nm una de la otra.

Una red que use DWDM con canales λ en m longitudes de onda, λ1, λ2, , λm, puede ser vista a nivel de conexión punto a punto como m copias de la red, cada una con la misma topología física, como se muestra en la figura 9. Una conexión óptica entre dos estaciones usa un camino óptico en la copia de la red correspondiente al canal λ asignado a esa conexión, es decir en una red óptica un camino que comunique a emisor-destino debe usar el mismo canal λ para todos los enlaces del camino.

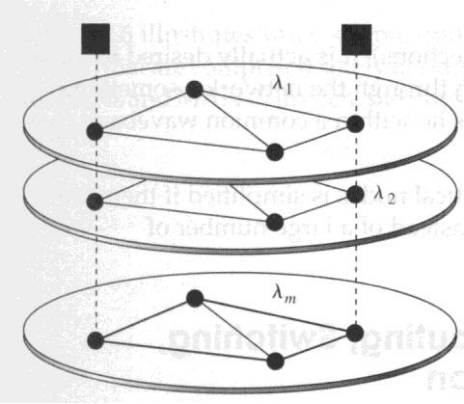


Figura 17 Imagen ilustrativa de una red multiplexada por longitud de onda

Fuente: STERN, Thomas y BALA Krishna. Multiwavelength Optical Networks a Layered Approach, New Jersey, USA : Prentice Hall 2000.

- **Arquitectura Multiprotocol Lambda Switching (MP λS).** La arquitectura MP λS es una arquitectura diseñada para redes ópticas por la IETF (Internet Engineering Task Force), está basada en MPLS la cual consiste en conmutar por medio de etiquetas que se le agregan a cada paquete, haciendo que la etiqueta sea un índice, que al buscarlo en una tabla sea posible encontrar la línea correcta de salida para el paquete.

MP λS trata de correlacionar una etiqueta MPLS a una longitud de onda específica, de tal forma de que los nodos ópticos no necesiten procesar la etiqueta MPLS sino que puedan tomar decisiones basadas en las longitudes de onda. Esto permite poder realizar

conmutación sin tener que realizar la conversión de la señal óptica a una señal eléctrica debido a que no habría necesidad de leer el contenido del paquete.

En la figura 18 podemos ver una gráfica del principio básico en que está basada la conmutación con MP λS, los diferentes colores de las flechas, indican diferentes longitudes de onda.

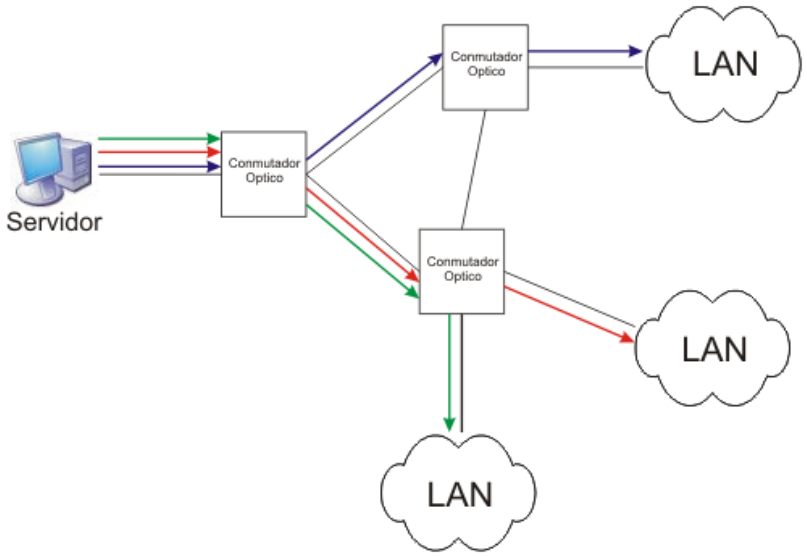


Figura 18 Ejemplo de conmutación lambda

2.6.4 Fibra Óptica. Como lo mencionamos anteriormente, no todas las redes que usan fibra óptica son redes ópticas, pero todas la redes ópticas usan fibra óptica es por esto que a continuación explicaremos cómo se propaga la luz a través de la fibra óptica.

En las comunicaciones ópticas, la fibra es simplemente el medio de transmisión lo que transmite es un pulso de luz generado por el transmisor óptico donde se indica el bit 1 si se presenta el pulso y bit 0 en ausencia de él.

Las fibras ópticas funcionan con el principio de reflexión de la luz, Tanembaum afirma: "Cualquier rayo de luz que incida en la frontera con un ángulo mayor que el crítico se

reflejará internamente, muchos rayos estarán rebotando dentro de la sílice con ángulos diferentes"[30]. Así pues, la fibra se propaga llevando la información de un extremo a otro, gracias al principio de la reflexión interna total.

Como se puede ver en la figura 19 la fibra óptica está compuesta por tres partes, un núcleo de vidrio a través del cual se propaga la luz, el cual está cubierto por un revestimiento de vidrio con un índice de refreacción menos que el del núcleo, con el fin de mantener toda la luz en el núcleo, y por último, el revestimiento está cubierto a su vez por una cubierta plástica para protegerlo.

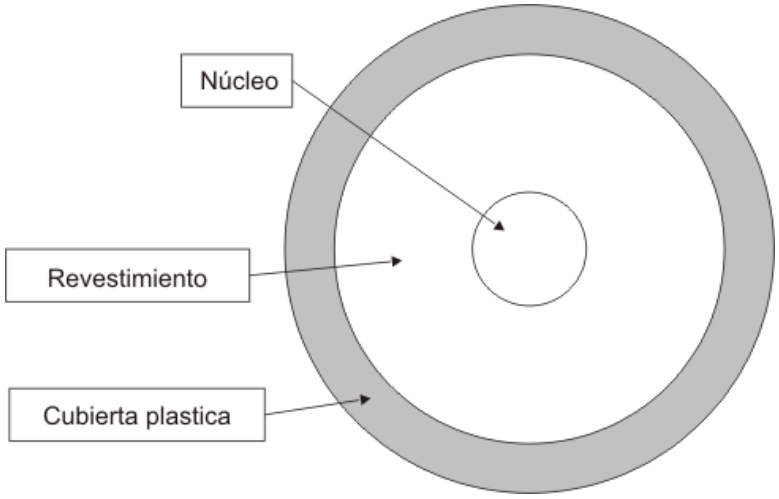


Figura 19 Fibra óptica desde una vista transversal

- **Tipos de fibras.** Existen tres tipos de fibra óptica dependiendo del diámetro del núcleo, del índice de refracción del núcleo. En cada tipo de fibra de luz viaja de manera distinta. Los diámetros del núcleo son medidos en micrones (un micrón es un micrómetro, que es la millonésima parte de un metro).

  - **Fibra multimodo step index.** En las fibras multimodo, el diámetro del núcleo varía de 50 micrones a 100 micrones. Además los rayos de luz entran por un

[30] Tanenbaum, Andrew S. Redes de computadores. Tercera edición. México: Prentice Hall Hispanoamérica, S.A. 1997

número finito de ángulos y viajan por un número finito de caminos. Los otros caminos posibles interactúan unos con otros y se destruyen en el proceso de interferencia. Se le llama step index, debido a que el índice de refracción del núcleo es constante.

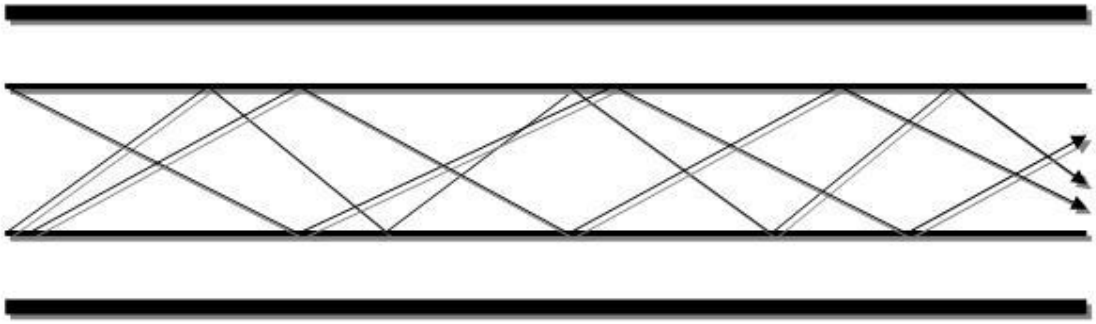


Figura 20 Fibra multimodo step index

Fuente: Mukherjee, Biswanath. Optical Communication Networks. California U.S.A : McGraw-Hill 1.997.

En la figura 20 se puede ver cómo viaja la luz en estas fibras, y en la figura 21 se puede ver como es el índice de refracción con respecto al radio de la fibra, en donde *a* es el radio del núcleo y *b*, es el radio del revestimiento de vidrio.

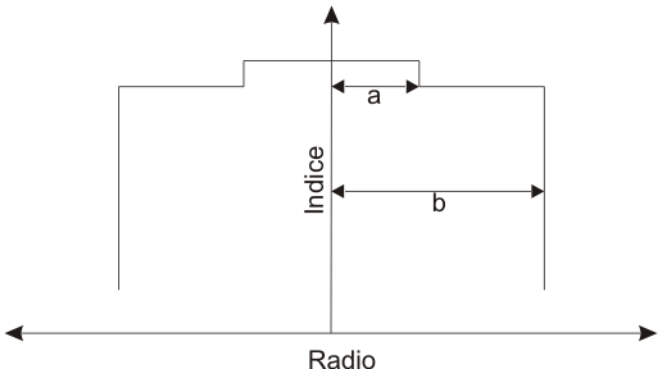


Figura 21 Gráfica del radio contra el índice de refracción de la fibra step index.

- **Fibra multimodo graded index.** Por ser multimodo posee las mismas características que la anterior fibra en lo que corresponde a diámetro del núcleo y en

que los rayos de luz entran por diferentes ángulos, la diferencia entre ésta y la fibra anterior es que esta es graded index lo que significa que el núcleo tiene un índice de refracción variable de tal manera, que entre más cerca al centro se esté, mayor será el índice de refracción del núcleo.

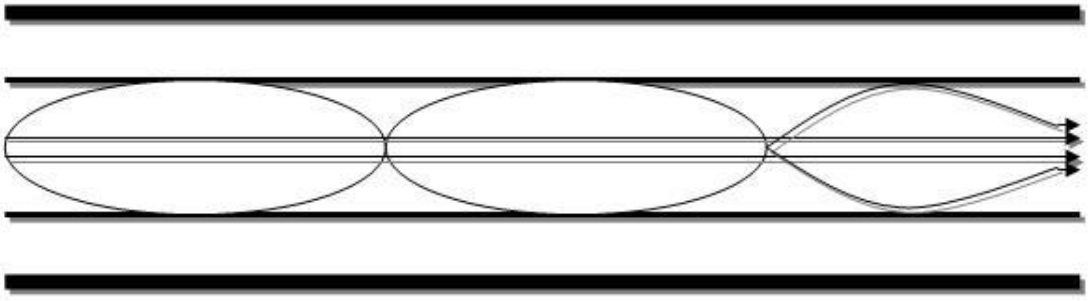


Figura 22 Fibra óptica multimodo graded index

Fuente: Mukherjee, Biswanath. Optical Communication Networks. California U.S.A : McGraw-Hill 1.997.

En la figura 22 se puede ver cómo viaja la luz en estas fibras, y, en la figura 23 se puede ver cómo es el índice de refracción con respecto al radio de la fibra en donde *a* es el radio del núcleo y *b* es el radio de revestimiento de vidrio.

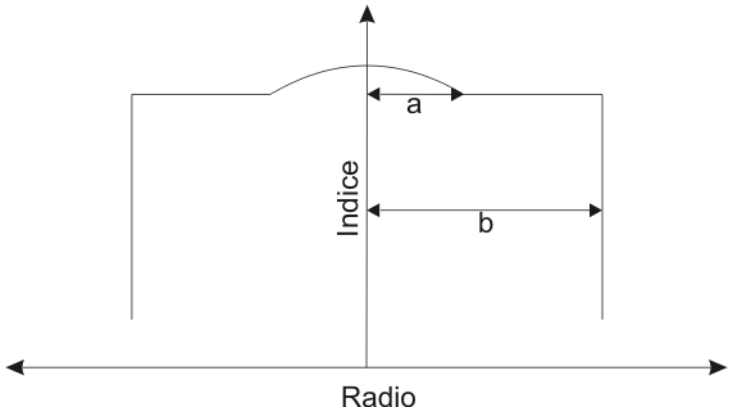


Figura 23 Gráfica del radio contra el índice de refracción de la fibra graded index.

- **Fibra monomodo step index.** En estas fibras el diámetro del núcleo va de 7 a 9 micrones, lo cual es considerablemente más delgado que las fibras multimodo, debido a estos es que en estas fibras la luz no rebota, sino que la luz viaja en línea recta de manera que solo pasa un rayo de luz, lo que produce poca distorsión.

En la figura 24 se puede ver cómo viaja la luz en estas fibras.

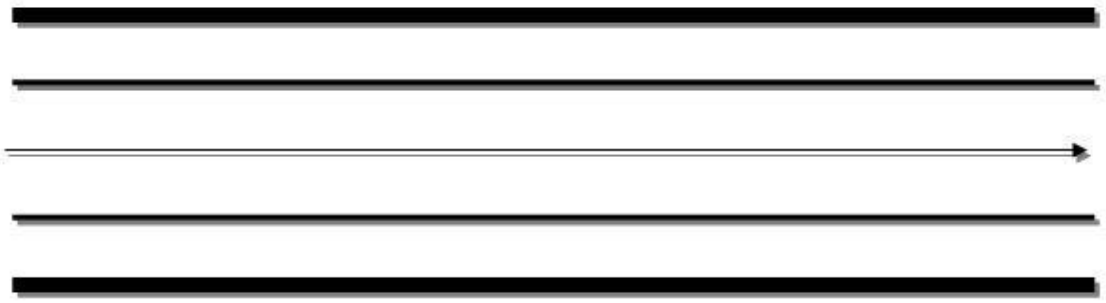


Figura 24 Fibra óptica monomodo graded index.

Fuente: Mukherjee, Biswanath. Optical Communication Networks. California U.S.A : McGraw-Hill 1.997.

3. DISEÑO DEL ALGORITMO HÍBRIDO ANT COLONY-EVOLUTIVO

3.1 MODELO MATEMATICO

En concordancia con[31], la red donde se realizará la optimización es modelada por un grafo $G = (N, E)$ en el cual el conjunto de nodos es denotado por la letra N y el conjunto de enlaces por la letra E. Se considera a T como el conjunto de nodos egresos, $T \subset N$. El nodo origen en la transmisión está denotado por $s \in N$ (nodo ingreso) y $t \in T$ puede ser cualquier nodo destino. El enlace existente entre el nodo i y el nodo j se denota como (i, j), teniendo en cuenta que $(i, j) \in E$. Sea Λ el conjunto de lambdas, la variable $\lambda \in \Lambda$ indica la longitud de onda de la fibra utilizada para la transmisión en el enlace $(i, j) \in E$. La letra f representa el flujo Multicast, $f \in F$ que es el conjunto de flujos y T_f denota los nodos destinos subconjunto del flujo Multicast.

El objetivo del modelo de optimización a plantear es minimizar al mismo tiempo:

- El número de longitudes de ondas utilizadas.
- La atenuación máxima en un enlace.
- El retardo. Para lo cual se tiene en cuenta el retardo total, retardo promedio, retardo máximo y la máxima variación del retardo.
- El número de saltos. En donde se debe considerar número de saltos total, promedio de número de saltos, máximo número de saltos y la máxima variación del número de saltos.
- Ancho de banda consumido.

[31] Op. Cit. p 47

Con este fin, se define la variable $X_{ij}^{\lambda ft}$ como una variable binaria la cual representa la presencia o no de información en el árbol multicast *t* del flujo *f* en la longitud de onda λ en el enlace (*i, j*). En otras palabras:

$$X_{ij}^{\lambda ft} \in Z, [0,1] = \begin{cases} 1, & \text{si el flujo f es trasmitido en } \lambda \text{ por (i, j)} \in \text{E con destino t.} \\ 0, & \text{en otro caso} \end{cases} \qquad \textbf{(1)}$$

A continuación ampliamos los conceptos que están involucrados en el modelo matemático:

Longitudes de Onda

A partir de la variable $X_{ij}^{\lambda ft}$ se construye la ecuación para el primer objetivo a minimizar, para esto se tienen en cuenta sólo los enlaces utilizados (aquellos para los que el valor de *X* es uno), razón por la cual se utiliza la función *max*. Además, el número de longitudes de onda utilizadas se calcula para todo flujo y todos los destinos del árbol multicast.

$$\sum_{\lambda} \sum_{(i,j) \in E} \max(X_{ij}^{\lambda ft})_{ft} \qquad \textbf{(2)}$$

Atenuación

La atenuación en la fibra óptica es la reducción de la potencia de la señal, a medida que ésta se propaga a lo largo de la fibra. Con el fin de conocer cuál es la distancia máxima que puede recorrer una señal para que ésta pueda ser detectada por el receptor, se debe tener en cuenta la sensibilidad de éste último, en otras palabras, conocer la mínima potencia requerida para el reconocimiento de la señal.

Si *L* es la distancia en kilómetros (km) entre dos nodos comunicados a través de fibra óptica y *A* la constante de atenuación (la cual depende del tipo de fibra que se esté utilizando) dada en dB/Km; entonces, la potencia requerida es:

$$P(L) = 10^{-A_{ij} * L_{ij} / 10} * P(i) \qquad \textbf{(3)}$$

donde *P(i)* es la potencia del nodo emisor de la señal.

El segundo objetivo a minimizar es la Atenuación. En este caso se desea minimizar la máxima atenuación de un flujo dado debido a que en un árbol multicast, la atenuación puede variar entre enlaces. La ecuación:

$$\max\left[\left(10^{-A_{ij} * L_{ij} / 10} * P(i)\right) * \max\left(X_{ij}^{\lambda ft}\right)_t\right]_{f,\lambda,(i,j)} \qquad \textbf{(4)}$$

representa la atenuación en la fibra en el enlace (*i, j*) hacia el destino *t.*

Retardo

En una red, el retardo es una expresión de, -qué tanto tiempo le toma a un paquete viajar desde un nodo origen hasta su destino-.

Existen varios tipos de retardo:

- **Retardo de Colas.** Cada enrutador de una red toma un poco de tiempo para examinar el paquete que recibe y cambiar la cabecera del mismo; ésto produce un poco de represamiento en el dispositivo a medida que llegan más paquetes.

  En la investigación, este retardo no se toma en cuenta debido a que se está evaluando el proceso de transmisión multicast en redes conmutadas, donde sólo hay dos enrutadores (entrada y salida).

- **Retardo de Conmutación.** En cada conmutador que se encuentra en una red óptica, se requiere que se convierta la información Óptico – Eléctrico, con el fin de que se pueda establecer el puerto de reenvío de la información y una segunda conversión Eléctrico – Óptico, para el reenvío en sí.

  En la investigación, se considera este tiempo despreciable, debido a que la red de estudio está conformada por conmutadores OOO (óptico – óptico – óptico), que no requieren las conversiones descritas anteriormente.

- **Retardo de Transmisión.** En una red óptica, consiste en el tiempo que se toma un paquete para viajar de un sitio a otro a la velocidad de la luz. Éste es el retardo que se tiene en cuenta en la investigación, el cual está afectado directamente por la distancia entre dos nodos.

- **Retardo Jitter – Buffer.** Es el retardo que experimentan las aplicaciones multimedia de tiempo real; que en ocasiones repercute en la entrega no continua o fluida de la información. Con el fin de evitar esto, se adiciona un buffer, en el que se almacenan los paquetes recibidos y la tasa de entrega al nivel superior y se ajusta de acuerdo al rendimiento de la red.

  Debido a que este tipo de retardo afecta a un determinado grupo de aplicaciones, no se tendrá en cuenta en el modelo.

Para la minimización del retardo se tienen en cuenta el retardo total, el retardo promedio, el retardo máximo y la máxima variación del retardo. La minimización de cada una de estas medidas conduce a la minimización del retardo en general, debido a que en un camino, se puede encontrar un enlace con el menor retardo y, sin embargo, el camino puede hacer

parte del árbol multicast con el máximo retardo promedio o con la máxima variación del mismo, haciendo esta solución poco deseable.

El retardo total se calcula con la ecuación:

$$\sum_{f}\sum_{\lambda}\sum_{t}\sum_{(i,j)\in E}\left(d_{ij} * \left(X_{ij}^{\lambda ft}\right)\right) \qquad \textbf{(5)}$$

Donde $d_{i,j}$ representa la distancia entre el nodo *i* y el nodo *j*.

Por otra parte, el retardo promedio está dado por el retardo total dividido por la norma del conjunto de destinos de cada flujo; en otras palabras, se calcula el retardo promedio del árbol de transmisión multicast por flujo. La ecuación para el retardo promedio es la siguiente:

$$\frac{\sum_{f}\sum_{\lambda}\sum_{t}\sum_{(i,j)\in E}\left(d_{ij} * \left(X_{ij}^{\lambda ft}\right)\right)}{\sum_{f}\left|T_f\right|} \qquad \textbf{(6)}$$

También, ha de considerarse el máximo retardo experimentado por un flujo en el árbol de transmisión multicast debido a que afecta directamente la calidad de la información que viaja por ese flujo; es por eso que se plantea la ecuación

$$\max\left[\sum_{(i,j)\in E}\left(d_{ij} * \left(X_{ij}^{\lambda ft}\right)\right)\right]_{\lambda, f, t} \qquad \textbf{(7)}$$

Una medida adicional a minimizar en el modelo de optimización, es la máxima variación del retardo puesto que en el mismo árbol se podría encontrar tanto el mínimo, como el máximo retardo. La siguiente ecuación nos permite obtener dicha medida.

$$\left(\max\left[\sum_{(i,j)\in E} \left(d_{ij} * \left(X_{ij}^{\lambda ft} \right) \right) \right]_{\lambda, f, t} \right) - \left(\min\left[\sum_{(i,j)\in E} \left(d_{ij} * \left(X_{ij}^{\lambda ft} \right) \right) \right]_{\lambda, f, t} \right) \qquad \textbf{(8)}$$

Número de Saltos (f_2)

Otro de los objetivos a minimizar es el número de saltos; al igual que en el retardo, ha de considerarse el número de saltos total, el promedio de número de saltos, el máximo número de saltos y la máxima variación del número de saltos. Siguiendo un razonamiento similar a la obtención de las ecuaciones para la optimización del retardo, se tiene:

Número de saltos total, representa la cantidad de nodos por los que pasa un flujo, que viaja en una longitud de onda a través de en un árbol de transmisión multicast para alcanzar un destino *t*

$$\sum_{f} \sum_{\lambda} \sum_{t} \sum_{(i,j)\in E} \left(X_{ij}^{\lambda ft} \right) \qquad \textbf{(9)}$$

Promedio de numero de saltos, número de saltos total dividido por la norma del conjunto de destinos de cada flujo

$$\frac{\sum_{f} \sum_{\lambda} \sum_{t} \sum_{(i,j)\in E} \left(X_{ij}^{\lambda ft} \right)}{\sum_{f} \left| T_f \right|} \qquad \textbf{(10)}$$

Máximo número de saltos:

$$\max\left[\sum_{(i,j)\in E}\left(X_{ij}^{\lambda ft}\right)\right]_{\lambda,f,t} \qquad \textbf{(11)}$$

Máxima variación del número de saltos:

$$\left(\max\left[\sum_{(i,j)\in E}\left(X_{ij}^{\lambda ft}\right)\right]_{\lambda,f,t}\right)-\left(\min\left[\sum_{(i,j)\in E}\left(X_{ij}^{\lambda ft}\right)\right]_{\lambda,f,t}\right) \qquad \textbf{(12)}$$

Ancho de Banda (f_3)

Como última función objetivo del modelo se desea minimizar el ancho de banda consumido, el cual se representa mediante la siguiente ecuación:

$$\sum_{f}\sum_{\lambda}\sum_{(i,j)\in E}\left(b_f * \max\left(X_{ij}^{\lambda ft}\right)_t\right) \qquad \textbf{(13)}$$

Donde b_f representa el ancho de banda del flujo.

Modelo Matemático

De los ítems anteriores se obtiene un modelo con once objetivos a minimizar (*k = 11*), que están sujetas a *m=5* restricciones, en donde se observan tan sólo cinco funciones independientes: longitud de onda, atenuación, retardo, número de saltos y ancho de banda.

Entre las restricciones se encuentran las leyes de Kirchoff, las cuales garantizan la conservación del flujo y aseguran, adaptadas para este caso, que:

Flujo total que sale desde el nodo fuente a cualquier destino es igual a 1.

$$\sum_{(i,j)} X_{ij}^{\lambda ft} - \sum_{(j,i)} X_{ji}^{\lambda ft} = 1$$
$$\forall t \in T_f, \forall f \in F, \forall \lambda \in \Lambda, i = s \qquad \textbf{(14)}$$

- Flujo total que viene de cualquier destino es igual a 1.

$$\sum_{(i,j)} X_{ij}^{wft} - \sum_{(j,i)} X_{ji}^{wft} = 1$$
$$\forall i, t \in T_f, \forall f \in F, \forall \lambda \in \Lambda \qquad \textbf{(15)}$$

En los nodos intermedios el flujo de entrada es igual al flujo de salida.

$$\sum_{(i,j)} X_{ij}^{\lambda ft} - \sum_{(j,i)} X_{ji}^{\lambda ft} = 0$$
$$\forall t \in T_f, \forall f \in F, i \neq s, i \neq T_f \qquad \textbf{(16)}$$

Otra restricción tiene que ver con el límite del ancho de banda consumido, el cual debe ser menor o igual a la capacidad en el enlace (*i, j*). Además, la cantidad de longitudes de onda debe ser menor o igual que el número máximo disponibles por la fibra utilizada en el enlace (*i, j*). Así:

$$\sum_{w} \max(X_{ij}^{\lambda ft})_f \leq M\_X_{ij}, \forall (i, j) \in E \qquad \textbf{(17)}$$

Donde M_X_{ij} es el número máximo de longitudes de onda permitido en el enlace *(i,j)* $\in E$.

La variable C_{ij}^{λ} que representa la capacidad del enlace *(i,j)* transmitida por la longitud de onda λ; $b\lambda_{\lambda}$ es ancho de banda utilizado en la longitud de onda λ; M_X_{ij} es el número máximo de longitudes de onda permitido en el enlace *(i,j)* $\in E$.

El ancho de banda consumido se convierte en una restricción más del modelo.

$$\sum_{f} \max(X_{ij}^{\lambda ft})_t * b\lambda_\lambda \leq C_{ij}^{\lambda}, \forall (i,j) \in E, \lambda \in \Lambda \qquad \textbf{(18)}$$

La anterior ecuación, restringe la suma de los máximos anchos de banda $b\lambda_\lambda$ trasmitidos por la longitud de onda λ del enlace $(i,j) \in E$. La variable C_{ij}^{λ} que representa la capacidad del enlace *(i, j)* transmitida por la Lambda λ *y* $b\lambda_\lambda$ representa el ancho de banda utilizado en la Lambda λ*;* El resultado de la sumatoria debe ser menor o igual a la capacidad del enlace C_{ij}^{λ}.

Luego entonces, el problema de optimización multiobjetivo en redes ópticas con transmisión multicast puede formularse como:

Minimizar

$$f_1(X) = \sum_{f}\sum_{\lambda}\sum_{t}\sum_{(i,j)\in E}\left(d_{ij} * \left(X_{ij}^{\lambda ft}\right)\right)$$

$$f_2(X) = \sum_{f}\sum_{\lambda}\sum_{t}\sum_{(i,j)\in E}\left(X_{ij}^{\lambda ft}\right)$$

$$f_3(X) = \sum_{f}\sum_{\lambda}\sum_{(i,j)\in E}\left(b_f * \max\left(X_{ij}^{\lambda ft}\right)_t\right)$$

$$f_4(X) = \max\left[\sum_{(i,j)\in E}\left(d_{ij} * \left(X_{ij}^{\lambda ft}\right)\right)\right]_{\lambda,f,t}$$

$$f_5(X) = \max\left[\sum_{(i,j)\in E}\left(X_{ij}^{\lambda ft}\right)\right]_{\lambda,f,t}$$

$$f_6(X) = \frac{\sum_{f}\sum_{\lambda}\sum_{t}\sum_{(i,j)\in E}\left(d_{ij} * \left(X_{ij}^{\lambda ft}\right)\right)}{\sum_{f}\left|T_f\right|}$$

$$f_7(X) = \frac{\sum_f \sum_\lambda \sum_t \sum_{(i,j)\in E} \left(X_{ij}^{\lambda ft}\right)}{\sum_f |T_f|}$$

$$f_8(X) = \sum_\lambda \sum_{(i,j)\in E} \max(X_{ij}^{\lambda ft})_{ft}$$

$$f_9(X) = \left(\max\left[\sum_{(i,j)\in E}\left(d_{ij} * \left(X_{ij}^{\lambda ft}\right)\right)\right]_{\lambda,f,t}\right) - \left(\min\left[\sum_{(i,j)\in E}\left(d_{ij} * \left(X_{ij}^{\lambda ft}\right)\right)\right]_{\lambda,f,t}\right)$$

$$f_{10}(X) = \left(\max\left[\sum_{(i,j)\in E}\left(X_{ij}^{\lambda ft}\right)\right]_{\lambda,f,t}\right) - \left(\min\left[\sum_{(i,j)\in E}\left(X_{ij}^{\lambda ft}\right)\right]_{\lambda,f,t}\right)$$

$$f_{11}(X) = \max\left[\left(10^{-A_{ij} * L_{ij}/10} * P(i)\right) * \max\left(X_{ij}^{\lambda ft}\right)_t\right]_{f,\lambda,(i,j)}$$

Sujeto a

$$\sum_{(i,j)} X_{ij}^{\lambda ft} - \sum_{(j,i)} X_{ji}^{\lambda ft} = 1$$

$$\forall t \in T_f, \forall f \in F, \forall \lambda \in \Lambda, i = s$$

$$\sum_{(i,j)} X_{ij}^{\lambda ft} - \sum_{(j,i)} X_{ji}^{\lambda ft} = 1$$

$$\forall i, t \in T_f, \forall f \in F, \forall \lambda \in \Lambda$$

$$\sum_{(i,j)} X_{ij}^{\lambda ft} - \sum_{(j,i)} X_{ji}^{\lambda ft} = 0$$

$$\forall t \in T_f, \forall f \in F, i \neq s, i \neq T_f$$

$$\sum_w \max(X_{ij}^{\lambda ft})_f \leq M\_X_{ij}, \forall (i,j) \in E$$

$$\sum_f \max(X_{ij}^{\lambda ft})_t * b\lambda_\lambda \leq C_{ij}^\lambda, \forall (i,j) \in E, \lambda \in \Lambda$$

donde

$$X_{ij}^{\lambda ft} \in Z, [0,1] = \begin{cases} 1, & \text{si el flujo f es trasmitido en } \lambda \text{ por el (i, j)} \in \text{E con destino t.} \\ 0, & \text{en otro caso} \end{cases}$$

Los problemas computacionales pueden ser de distinta categoría como son:

- P, son aquellos que pueden ser resueltos en tiempo polinomial.
- NP, son problemas para los cuales no existen algoritmos para resolverlos con una complejidad de tiempo polinomial, pero se pueden aproximar a la solución con la utilización de algoritmos P.
- NP-completo, que a su vez son NP y para éste existe un algoritmo P para reducir el problema otro P.
- NP-Hard, problema de optimización en el cual se utiliza un problema NP-completo como subproceso.

El problema presentado es de naturaleza NP-Hard, ya que la minimización de las funciones objetivos: Atenuación, Retardo y número de longitudes de onda para un grupo Multicast se conoce como el problema del árbol de Steiner (NP-completo) cuyo modelo incluye restricciones en sus variables enteras y reales.

3.2 ALGORITMO HÍBRIDO ANT COLONY-EVOLUTIVO

El algoritmo hibrido diseñado para resolver el problema planteado en 3.1, lo llamaremos por sus siglas como MOANCOLE (Multi-objective Ant Colony Evolutionary), en este algoritmo las hormigas, no son consideradas iguales como sucede en el algoritmo Ant Colony, sino que las hormigas tienen asociado un fitness o desempeño, que induce a una diferencia entre ellas, al realizar la selección y cruzamiento para obtener el frente de Pareto, el algoritmo principal es el siguiente:

```
Procedure MOANTCOLE Metaheuristic
        For(i=0;i<num_gen;i++)
                constructAntSolutions()
                calculateFitness()
                EnviromentalSelection()
```

```
            selection()
            crossover()
            selectParetoFront()
            updatePheromoneTrail()
            pheromoneEvaporation()
    End For
End Procedure
```

Construct Ant Solutions

Esta fase del algoritmo se encarga de construir la colonia de hormigas

Calculate Fitness

Ver descripción del algoritmo SPEA 2.

Enviromental Selection

Ver descripción del algoritmo SPEA 2.

Selection

Ver descripción del algoritmo SPEA 2.

Crossover[32]

El operador de cruzamiento del algoritmo planteado, utiliza cruzamiento de árboles y cruzamiento de caminos.

El cruzamiento de árboles recibe dos árboles del mismo flujo que contengan los mismos destinos y a partir de ellos, se generan dos árboles hijos mediante el intercambio de caminos. La escogencia de la cantidad de caminos y cuáles caminos intercambiar en los árboles padres, se realiza en forma aleatoria con distribución uniforme. Por ejemplo, en el caso en que se aplicara el operador al árbol f1 y al árbol f2, los cuales son del mismo flujo,

[32] Op. Cit p 47

donde ambos tienen como destinos los nodos 1, 2, 3 y 4, se escoge de forma aleatoria intercambiar los caminos que tienen como destino a 2 y a 3, dando como resultado del cruzamiento los árboles f3 y f4. Una representación grafica del intercambio lo podemos ver en la figura 25.

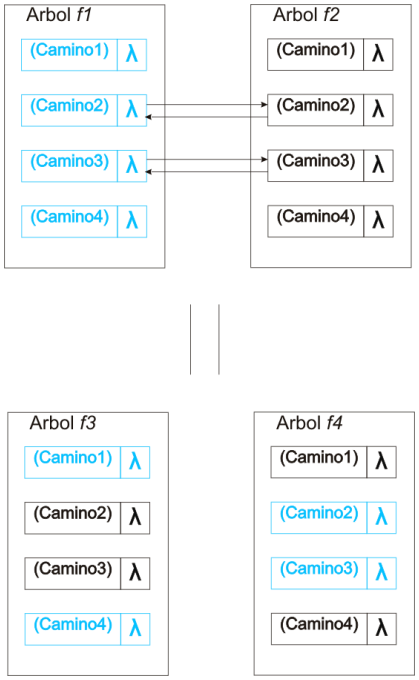


Figura 25 Ejemplo de cruzamiento de dos árboles.

Estas hormigas generadas se optimizan localmente utilizando un cruzamiento de caminos[33], basándose en los siguientes conceptos: *Punto de Intercepción y función Distancia de Intercepción.*

[33] DONOSO, Yesid, PEREZ, Alfredo, ARDILA, Carlos. Optimizing Multiple Objectives on Multicast Networks using Memetic Algorithms. GETS International transaction on Computer Science and Engineering. Seul, Korea. October 2005.

Definición

Sea $G=(V,E)$ un grafo, X e Y dos caminos acíclicos cualesquiera entre dos vértices de G y x un vértice, $x \in V$. Se dice que x es un *Punto de Intercepción* entre los caminos X e Y, si x se encuentra en los caminos X y Y. El *Punto de Intercepción* se denotará como $x_{XY}^{\angle}$.

Se denota como I_{XY} al conjunto de puntos intersección entre dos caminos X y Y.

$$I_{XY} := \{ x \in V \mid x_{XY}^{\angle} \} \qquad \textbf{(19)}$$

Esta heurística asume que el origen de los caminos es el mismo. En nuestro contexto significa que aplicaremos la heurística en cada árbol de las hormigas hijas. La figura 26 muestra dos caminos de un mismo árbol, sus puntos de intercepción y dos nuevos caminos creados a partir del cruzamiento de los caminos en dos puntos de intercepción.

Camino1, Destino: nodo 5

0	8	13	12	1	24	14	21	23	20	17	16	9	18	19	10	22	5

Camino2, Destino: nodo 7

0	19	2	13	3	11	10	16	22	5	12	1	20	6	8	21	14	4	7

$$I_{Camino1,Camino2}=\{0,19,13,10,16,22,5,12,1,20,8,21,14\}$$

Nuevos Caminos creados a partir de los puntos de intercepción 19 y 14.

$x^{\angle}_{ca\min o1,ca\min o2}$ = 19, nuevo camino creado por cruzamiento entre el Camino2 y Camino1.

Camino 3, Destino: nodo 5

0	19	10	22	5

$x^{\angle}_{ca\min o1,ca\min o2}$ = 14, nuevo camino creado por cruzamiento entre el Camino1 y Camino2.

Camino 4, Destino: nodo 7

0	8	13	12	1	24	14	4	7

Figura 26 Puntos de intercepción entre dos caminos. Por cada punto de intercepción distinto al origen, se podrían formar nuevos caminos aplicando un cruzamiento entre los caminos en los puntos de intercepción.

Definición

Sea $x^{\angle}_{XY}$ un punto de intercepción, $\text{Pos}_X(x^{\angle}_{XY})$, $\text{Pos}_Y(x^{\angle}_{XY})$ la función que retorna la posición del punto de intercepción $x^{\angle}_{XY}$ en el camino *X* y el camino *Y* respectivamente (recuérdese que un camino es un vector de enteros). Se define la *función Distancia de Intercepción* como:

$$D(x^{\angle}_{XY}) := \text{abs}(\text{Pos}_X(x^{\angle}_{XY}) - \text{Pos}_Y(x^{\angle}_{XY})) \qquad \textbf{(20)}$$

donde abs(*w*) denota la función valor absoluto.

Esta definición, nos permitirá escoger el punto de cruzamiento para los caminos. Se escogerá como punto de cruzamiento, aquel punto tal, que su $D(x_{XY}^{\angle})$ es máxima en el conjunto $I_{XY.}$ Si $\forall\, x_{XY}^{\angle} \in I_{XY,}\, D(x_{XY}^{\angle}) = 0$, entonces se escoge $x_{XY}^{\angle}$ tal que su $Pos_X(x_{XY}^{\angle})$ es la máxima. Si existen varios puntos con $D(x_{XY}^{\angle})$ máxima en I_{XY}, se puede escoger cualquiera de esos puntos.

Escogido $x_{XY}^{\angle}$, se obtienen los subcaminos desde el origen hasta $x_{XY}^{\angle}$ en los caminos *X* e *Y*. Sean X_1 *e* Y_1 dichos subcaminos. Si $X_1 \sim Y_1$, el algoritmo no crea un nuevo camino. Si $X_1 \prec Y_1$ entonces se obtiene el subcamino desde $x_{XY}^{\angle}$ hasta el nodo destino en el camino *Y*, este nuevo subcamino se denota como Y_2 y el nuevo camino se crea uniendo los subcaminos X_1 con Y_2 en el punto de intercepción. Se realiza lo mismo si $Y_1 \prec X_{1,}$ pero en este caso se crea el camino uniendo Y_1 con $X_{2.}$

Este mismo procedimiento se realiza entre cada par de caminos de un árbol, y la hormiga optimizada localmente se obtiene al aplicar la heurística a todos sus árboles. La figura 27 muestra paso a paso lo que hace la heurística con un árbol.

Arbol Inicial
Camino1, Destino: nodo 5

Nodos	**0**	**8**	**13**	**12**	**1**	24	**14**	**21**	23	**20**	17	**16**	9	18	**19**	**10**	**22**	**5**
Posición	**0**	**1**	**2**	**3**	**4**	5	**6**	**7**	8	**9**	10	**11**	12	13	**14**	**15**	**16**	**17**

Camino2, Destino: nodo 7

Nodos	**0**	**19**	2	**13**	3	11	**10**	**16**	**22**	**5**	**12**	**1**	**20**	6	**8**	**21**	**14**	4	7
Posición	**0**	**1**	2	**3**	4	5	**6**	**7**	**8**	**9**	**10**	**11**	**12**	13	**14**	**15**	**16**	17	18

Paso 1. Búsqueda de los puntos de intersección.

$I_{Camino1,Camino2}$={0,19,13,10,16,22,5,12,1,20,8,21,14}

Paso 2. Cálculo de $D(X^{\angle}_{XY})$ y escogencia del punto de intersección

$X^{\angle}_{ca\min o1, ca\min o2}$	$Pos_{camino1}(X^{\angle}_{ca\min o1, ca\min o2})$	$Pos_{camino2}(X^{\angle}_{ca\min o1, ca\min o2})$	$D(X^{\angle}_{ca\min o1, ca\min o2})$
0	0	0	0
19	14	1	13
13	2	3	1
10	3	10	7
16	11	7	4
22	15	6	9
5	17	9	8
12	3	10	7
1	4	11	7
20	9	12	3
8	1	14	13
21	7	15	8
14	16	6	10

Punto de intersección seleccionado: 19

Paso 3. Obtención de subcaminos y cálculo de dominancia.

$Camino1_1$

0	8	13	12	1	24	14	21	23	20	17	16	9	18	19

$Camino2_1$

0	19

$Camino2_1 \prec Camino1_1$

Paso4. Obtención de nuevo camino.

$Camino1_2$

19	10	22	5

Nuevo Camino Generado: $Camino2_1\ Camino1_2$

0	19	10	22	5

Arbol después de la Búsqueda Local
Camino1, Destino: nodo 5

Nodos	0	19	10	22	5
Posición	0	1	2	3	4

Camino2, Destino: nodo 7

Nodos	0	19	2	13	3	11	10	16	22	5	12	1	20	6	8	21	14	4	7
Posición	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18

Figura 27 Método de Búsqueda local utilizado sobre un Individuo con dos caminos.

El uso de el punto de intercepción con mayor $D(x^{\angle}_{XY})$ para generar el nuevo camino junto a la dominancia en los subcaminos, garantizan que el nuevo camino no tenga ciclos. Para

afirmar lo anterior, se ha demostrado que el algoritmo generador de nuevos caminos no genera caminos con ciclos y dicha demostración es la siguiente:

Teorema: El algoritmo generador de caminos genera caminos sin ciclos[34].

Demostración: ***(reducción al absurdo)***

Supongamos que el algoritmo genera caminos con ciclos. Sea *Z* un camino de este tipo generado por el algoritmo.

Puesto que *Z* es un camino con ciclos, $\exists\ v \in Z$ (*v* es un nodo) tal que tiene dos posiciones en *Z*. Denotemos con p_1 y p_2 estas posiciones con $p_1 < p_2$. Según el algoritmo, *Z* es generado de otros dos caminos *X e Y*, con punto $x_{XY}^{\angle}$ de intercepción escogido de tal manera que $D(x_{XY}^{\angle})$ es máxima en el conjunto I_{XY} o si $D(x_{XY}^{\angle}) = 0 \ \ \forall\, x_{XY}^{\angle} \in I_{XY}$ entonces $Pos_X(x_{XY}^{\angle}) > Pos_X(g_{XY}^{\angle})$ con $g_{XY}^{\angle} \in I_{XY}$ y $g_{XY}^{\angle} \neq x_{XY}^{\angle}$.

Sin perder generalidad, *Z* es generado cruzando los caminos en $x_{XY}^{\angle}$ tomando el subcamino desde el origen hasta $Pos_X(x_{XY}^{\angle})$ (puesto que se asume que el subcamino desde el origen hacia $Pos_X(x_{XY}^{\angle})$ domina a el subcamino desde el origen hasta $Pos_Y(x_{XY}^{\angle})$) y el subcamino desde $Pos_Y(x_{XY}^{\angle})$ hasta el destino del camino *Y*, de aquí se tiene lo siguiente:

1. $Pos_X(v) = p_1 < Pos_X(x_{XY}^{\angle})$
2. $Pos_Y(x_{XY}^{\angle}) < Pos_Y(v)$
3. $Pos_X(x_{XY}^{\angle}) \leq Pos_Y(x_{XY}^{\angle})$ (*por la dominancia de los subcaminos*)

Sumando

$0 < Pos_X(v) + Pos_Y(x_{XY}^{\angle}) < Pos_X(x_{XY}^{\angle}) + Pos_Y(v)$

$0 \leq Pos_Y(x_{XY}^{\angle}) - Pos_X(x_{XY}^{\angle}) < Pos_Y(v) - Pos_X(v)$

$0 \leq -(Pos_X(x_{XY}^{\angle}) - Pos_Y(x_{XY}^{\angle})) < -(Pos_X(v) - Pos_Y(v))$

[34] Ibid 33.

$0 \leq$ abs(- ($Pos_X(x^{\angle}_{XY})$ - $Pos_X(x^{\angle}_{XY})$)) $<$ abs (- ($Pos_X(v)$ - $Pos_Y(v)$))

$0 \leq$ abs($Pos_X(x^{\angle}_{XY})$ - $Pos_X(x^{\angle}_{XY})$) $<$ abs ($Pos_X(v)$ - $Pos_Y(v)$)

$0 \leq D(x^{\angle}_{XY}) < D(v)$

Esto es un absurdo puesto que el algoritmo escoge siempre $x^{\angle}_{XY}$ con $D(x^{\angle}_{XY})$ máxima •.

Los algoritmos para el cruce de caminos y la búsqueda local, son los siguientes:

```
Funcion  generador_camino(Camino1,Camino2)
1. I_Camino1,Camino2=buscar_puntos_de_interseccion(camino1,camino2)
2. Si (I_Camino1,Camino2 ≠ ∅ ) entonces
3.        x=seleccionar_punto_intersección(I_Camino1,Camino2)
4.        camino1_1=obtener_subcamino(origen,x,camino1)
5.        camino2_1=obtener_subcamino(origen,x,camino2)
6.        Si(camino1_1≺ camino2_1 en las funciones salto y retardo)
7                 camino2_2=obtener_subcamino(x,destino(camino2),camino2)
8.                retornar camino1_1 camino2_2
9.        Sino
10.               Si(camino2_1≺ camino1_1 en las funciones salto y retardo)
11.                       camino1_2=obtener_subcamino(x,destino(camino1),camino2)
12.                       retornar camino2_1 camino1_2
13.               Sino
14                        retornar nulo.
15.               fin Si
16.       fin si
17.Sino
18.       retornar nulo.
19 fin Si
```

```
Procedimiento búsqueda local(individuo L)
1.Para(i=0,i<L.retornar_tamaño_arbol(),1)
2.        Para(j=i+1; L.retornar_tamaño_arbol(),1)
3                 nuevo_camino= generador_camino(L.retornar_camino(i), L.retornar_camino(j)
4.                Si(nuevo_camino ≠ nulo)
5.                        Si(destino(nuevo_camino)=destino(L.retornar_camino(i))
6.                                L.cambiar_camino(i,nuevo_camino)
7.                        Sino
8.                                L.cambiar_camino(j,nuevo_camino)
10.                       fin Si
11.               fin Si
12.       Fin Para
13.Fin Para
```

Figura 28 Algoritmo generador de caminos (figura superior) y Algoritmo de búsqueda local (figura inferior)..

Select Pareto Front

Esta etapa del algoritmo selecciona de la población de hormigas conformadas por hijos y padres, las hormigas no dominadas.

Update Pheromone Matriz

Con las hormigas seleccionadas en la etapa anterior, se actualiza la matriz de feromonas del problema.

Pheromone Evaporation

Esta etapa del algoritmo utiliza un factor de evaporación que evita que la matriz de feromonas se sature y así no permitir que las hormigas tomen los mismos caminos.

3.3 COMPLEJIDAD DEL ALGORITMO

Para el procedimiento Construct Ant Solutions() su complejidad viene dada por:

$$\Theta(n) = n_{ants} * \Theta(Construct\_Path) * n_{nodes}$$

Debido a que el peor de los casos se da cuando todos los nodos son destinos.

Pero:

$\Theta(Construct\_Path) = n^2{}_{nodos}$ porque se utiliza una búsqueda en profundidad y el grafo fue implementado mediante una matriz de adyacencia.

Luego obtenemos que el procedimiento Construct Ant Solutions() tiene complejidad $\Theta(n) = n_{nodos}{}^{3} * n_{ants}$.

Calcular Fitness

En el procedimiento calcular Fitness(), para cada hormiga, su complejidad viene dada por:

$$\Theta(n) = \Theta(R(i)) + \Theta(D(i)).$$

Como *R(i)* es la suma de los elementos que *i* domina, entonces en el peor de los casos *i* podría dominar n_{ants}*-1*, luego $\Theta(R(i)) = \Theta(n_{ants})$ para cada hormiga. Por otra parte, para el cálculo de *D(i)* se requiere ordenar la lista de individuos ascendentemente, lo cual es $\Theta(n_{ants} \log_2 n_{ants})$; además se debe calcular la distancia euclidiana desde un individuo hacia todos los demás, lo cual es $\Theta(n^2{}_{ants})$; entonces:

$$\Theta(D(i)) = \Theta(n^2{}_{ants}) + \Theta(n_{ants} \log_2(n_{ants})) \approx \Theta(n^2{}_{ants}).$$

Luego la complejidad teniendo en cuenta cada función objetivo es:

$\Theta(calcularFitness) = \Theta(n_{ants}) + \Theta(n^2{}_{ants}) \approx \Theta(n^2{}_{ants})$ para calcular el fitness de una hormiga.

Entonces, para hacer el cálculo del fitness de todas las hormigas:

$$\Theta(calcularFitness) = n^2{}_{ants} * n_{ants} \approx \Theta(n^3{}_{ants})$$

Para las 11 funciones objetivo:

$$\Theta(calcularFitness) = 11 * \Theta(n^3) \approx \Theta(n^3{}_{ants})$$

Para el procedimiento Enviromental Selection(), asumiendo el peor caso, el cual ocurre si consideramos a todas las hormigas como parte de la élite, la complejidad viene dada por $\Theta(enviroment) = \Theta(n_{ants}{}^2)$ debido a que es necesario confrontar cada hormiga contra las demás para verificar la dominancia.

El procedimiento *selection()*, donde se aplica torneo binario, es $\Theta(r * n_{antsElite}) \approx \Theta(n_{antsElite}) \approx \Theta(n_{ants})$, donde *r* es el número de enfrentamientos en el torneo binario y considerando que en el peor de los casos $n_{elite} = n_{ants}$.

Crossover

Para cruzar dos hormigas se aplicará el cruce de árboles. El peor caso ocurre cuando existen n_{nodos} - *1* nodos destinos, luego esto es $\Theta(cruceArboles) = n_{nodos}$. El procedimiento crossover hará $(k*n_{ants})/2$ cruces, donde $k \in (0,1)$. Luego la complejidad total es $\Theta\left(k/2 * n_{ants} * n_{nodos}\right) = \Theta\left(n_{ants} * n_{nodos}\right)$

Búsqueda Local (cruce de caminos)

En la búsqueda local, se recorren caminos buscando puntos de intersección entre un par de caminos, lo cual implica que en el peor de los casos los dos caminos tengan longitud $n_{nodos,}$ y la búsqueda de dichos puntos se realice comparando cada nodo del primer camino contra todos los del segundo camino. Este método tiene, por tanto $\Theta(n^2)$. Para abarcar todos los caminos:

$$\sum_{i=0}^{n}\sum_{j=i}^{n} n^2 = \sum_{j=i}^{n} n^2(n-i+1) = \sum_{j=i}^{n}(n^3 - n^2 i + n^2) = n^4 - n^2\sum_{j=i}^{n} i + n^3 = n^4 - n^2\left(\frac{n(n+1)}{2}\right) + n^3$$

$$= n^4 - n^2\left(\frac{n(n+1)}{2}\right) + n^3 = n^4 - \left(\frac{n^3(n+1)}{2}\right) + n^3 = n^4 - \frac{n^4}{2} - \frac{n^3}{2} + n^3 \approx \Theta(n^4{}_{nodos})$$

Donde *n* representa el número de nodos.

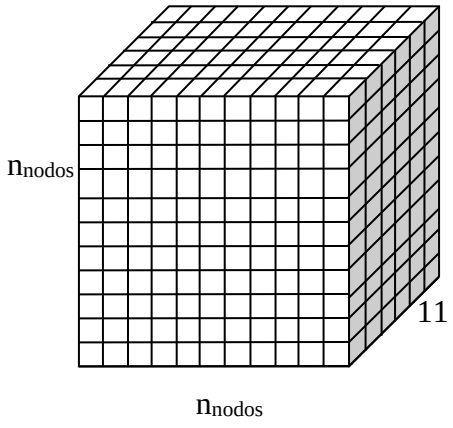
Select Pareto Front

Para hallar los elementos no dominados se necesita comparar cada hormiga con las demás para comprobar la dominancia, lo cual es $\Theta(n_{ants}{}^2)$. Pero esto debe hacerse en las 11 funciones: $\Theta(11 * n_{ants}{}^2) \approx \Theta(n_{ants}{}^2)$.

Update Pheromone Trail

Se actualizará la cantidad de feromona presente en cada enlace *i,j* de un camino correspondiente a cada función objetivo. La matriz presenta la estructura mostrada en la figura 29.

Figura 29. Cubo



Luego es claro que la complejidad está asociada a $\Theta(n_{nodos} * n_{nodos} * 11) \approx \Theta(n^2{}_{nodos})$

Pheromone Evaporation

Este procedimiento es análogo a Update Pheromone Trail, ya que recorrerá todos los elementos de la matriz de feromonas, luego es $\Theta(n^2_{nodos})$.

Complejidad del MOANTCOLE

$$\Theta(MOANTCOLE) = \Theta(n_{nodos}{}^3 * n_{ants}) + \Theta(n^3_{ants}) + \Theta(n_{ants}{}^2) + \Theta(n_{ants}) + \Theta(n_{ants} * n_{nodos}) + \Theta(n^4_{nodos}) + \Theta(n^2_{ants}) + \Theta(n^2_{nodos}) + \Theta(n^2_{nodos})$$

$$= \Theta(n^4_{nodos}) + \Theta(n^3_{ants}) + 2*\Theta(n^2_{nodos}) + 2*\Theta(n_{ants}{}^2) + \Theta(n_{nodos}{}^3 * n_{ants}) + \Theta(n_{ants} * n_{nodos}) + \Theta(n_{ants})$$

$\Theta(n_{ants})$ y $\Theta(n_{ants}{}^2)$ son absorbidos por $\Theta(n^3_{ants})$, $2*\Theta(n^2_{nodos})$ es absorbido por $\Theta(n^4_{nodos})$ y $\Theta(n_{nodos}{}^3 * n_{ants})$ absorbe a $\Theta(n_{ants} * n_{nodos})$ Por último, el término $\Theta(n_{nodos}{}^3 * n_{ants})$ es absorbido por $\Theta(n^4_{nodos}) + \Theta(n^3_{ants})$. Luego:

Entonces, para cada una de las generaciones del algoritmo:

$$\Theta(MOANTCOLE) = \Theta(n^4_{nodos}) + \Theta(n^3_{ants})$$

La complejidad definitiva de todas las iteraciones, teniendo en cuenta el número total de generaciones, viene dada por:

$$\Theta(MOANTCOLE) = n_{generaciones} * [\Theta(n^4_{nodos}) + \Theta(n^3_{ants})]$$

De lo anterior, podemos deducir que el algoritmo en el tiempo de ejecución se comporta polinominalmente, al funcionar sobre un problema NP Hard.

4. EXPERIMENTACION Y ANALISIS DE RESULTADOS

Para la experimentación, se realizó una prueba piloto constituida por cinco corridas (5) durante las cuales los algoritmos Ant Colony (MOANTCOL Multi Objective Ant Colony) y Ant Colony Evolutivo (MOANTCOLE Multi Objective Ant Colony Evolutive) se ejecutaron cien veces (100). Dependiendo de la convergencia de los algoritmos, la prueba piloto se aceptará o no como definitiva, esta justificación se presenta en la sección 4.2 del presente trabajo.

4.1 DESCRIPCIÓN DE LOS ARCHIVOS DE ENTRADA[35]

Para la realización del experimento se tomaron tres redes muy conocidas (National Science Foundation, Sprint y MCI), estableciendo un archivo de entrada con el siguiente formato:

- Cantidad de Nodos.
- Cantidad de Flujos.
- Potencia del Láser; la cual se encuentra en dBm.
- Cantidad de Lambdas disponibles y ancho de banda de cada una.
- Nodo origen y tamaño del flujo
- Nodos destinos del flujo, los cuales se eligen en forma aleatoria, donde todos tienen la misma probabilidad de ser seleccionados.

[35] CORONELL, Margarita y TOVAR, Luis, Optimización Multiobjetivo sobre Redes Ópticas en transmisión Multicast. Barranquilla 2006, 126 p. Trabajo de Grado (Maestría en Ciencias de la Computación). Instituto Tecnológico y de Estudios Superiores de Monterrey- Universidad Autónoma de Bucaramanga - Universidad Tecnológica de Bolívar

- Matriz de la red.
    - Nodo_i.
    - Nodo_j.
    - Retardo. El cálculo del retardo se basa en la distancia entre el nodo_i y el nodo_j, teniendo en cuenta que la velocidad de la luz en una fibra es 2 x 10^8 m/s.
    - Distancia. La distancia euclidiana entre el nodo_i y el nodo_j en cientos de kilómetros; basados en que cada 100 km se encuentran repetidores.
    - Factor de atenuación. Se encuentra entre 0.2 y 0.5 dB.

4.2 CARACTERÍSTICAS DEL EXPERIMENTO

El experimento toma las topologías establecidas, y se analiza el comportamiento, cuando el tráfico multicast se dirige hacia nodos destinos tomando el 30%, y 90% del total de nodos de la red.

Mientras la cantidad de los nodos destinos varían, los parámetros de entrada para iniciar la ejecución del algoritmo son los siguientes:

- Población: 100
- Archive Size: 50
- Probabilidad de Cruzamiento: 0. 85
- Rastro de feromona (inicial): 2.0
- Factor de evaporación para la matriz de feromona (fija) : 0.3

A continuación se muestran las tablas que describen los datos de entrada de cada topología seleccionada[36].

4.2.1 Topología NSF (National Science Foundation).

Ciudades	Nodo #	Potencia (mW)	Potencia (dBm)
Ann Arbor	0	2,4	3,8
Atlanta	1	6,4	8,1
Boulder	2	7,1	8,5
Champaign	3	5,8	7,6
College Pk	4	3,4	5,3
Houston	5	6	7,8
Ithaca	6	1,66	2,2
Lincoln	7	2	3
Palo Alto	8	3,8	5,8
Pittsburg	9	3,1	4,9
Princeton	10	5,8	7,6
Salt Lake City	11	3,4	5,3
San Diego	12	6	7,8
Seattle	13	1,66	2,2

Tabla 1 NSF - Nodos Topología - Potencia del Láser

Porcentaje	Origen	Destinos				
30%	3	5	6	11	12	
90%	7	0	2	4	8	10
	8	3	6	7	12	13
	10	1	3	11		

Tabla 2 NSF - Nodos Destinos

[36] Ibid 35.

Nodo_i	Nodo_j	Retardo (ms)	Distancia*	Atenuación (dB)	Distancia(Km)
0	6	2,8905	5,78	0,2	578,1
0	10	3,8425	7,69	0,2	768,5
0	11	11,458	22,92	0,2	2291,6
1	5	5,56	11,12	0,4	1112
1	9	4,0865	8,17	0,4	817,3
2	5	7,128	14,26	0,3	1425,6
2	7	3,573	7,15	0,3	714,6
2	11	2,7525	5,51	0,3	550,5
3	7	3,525	7,05	0,4	705
3	9	3,402	6,80	0,4	680,4
3	13	13,8145	27,63	0,4	2762,9
4	5	9,8545	19,71	0,2	1970,9
4	6	1,7375	3,48	0,2	347,5
4	10	0,9195	1,84	0,2	183,9
5	1	5,56	11,12	0,4	1112
5	2	7,128	14,26	0,4	1425,6
5	4	9,8545	19,71	0,4	1970,9
5	12	10,247	20,49	0,4	2049,4
6	0	2,8905	5,78	0,3	578,1
6	4	1,7375	3,48	0,3	347,5
6	9	1,781	3,56	0,3	356,2
7	2	3,573	7,15	0,4	714,6
7	3	3,525	7,05	0,4	705
8	11	4,599	9,20	0,2	919,8
8	12	3,3	6,60	0,2	660
8	13	5,7035	11,41	0,2	1140,7
9	1	4,0865	8,17	0,4	817,3
9	3	3,402	6,80	0,4	680,4
9	6	1,781	3,56	0,4	356,2
9	10	2,158	4,32	0,4	431,6
10	0	3,8425	7,69	0,3	768,5
10	4	0,9195	1,84	0,3	183,9
10	9	2,158	4,32	0,3	431,6
11	0	11,458	22,92	0,4	2291,6
11	2	2,7525	5,51	0,4	550,5
11	8	4,599	9,20	0,4	919,8
12	5	10,247	20,49	0,4	2049,4
12	8	3,3	6,60	0,4	660
12	13	8,4785	16,96	0,2	1695,7
13	3	13,8145	27,63	0,2	2762,9

13	8	5,7035	11,41	0,2	1140,7
13	12	8,4785	16,96	0,2	1695,7

Tabla 3 NSF – Matriz de la Red

4.2.2 Topología MCI.

Ciudades	Nodo #	Potencia (mW)	Potencia (dBm)
Austell	0	3	4,8
Charlton	1	5	7
Dallas	2	6	7,8
Denver	3	2,4	3,8
Greensboro	4	6,4	8,1
Houston	5	7,1	8,5
Independence	6	5,8	7,6
Los Ángeles	7	3,4	5,3
New York	8	6	7,8
Pompano Beach	9	1,66	2,2
Rialto	10	2	3
Sacramento	11	3,8	5,8
San Francisco	12	3,1	4,9
Seattle	13	9	9,5
W. Springs	14	2	3
Washington	15	1,68	2,3

Tabla 4 MCI F - Nodos Topología - Potencia del Láser

Porcentaje	Origen	Destinos				
30%	2	3	6	10	11	14
90%	15	1	4	7	11	12
	1	0	5	8	10	14
	5	3	9	13	14	

Tabla 5 MCI - Nodos Destinos

Nodo_i	Nodo_j	Retardo (ms)	Distancia*	Atenuación (dB)	Distancia (Km)	Tipo
0	2	5,6725	11,35	0,3	1134,5	OC-12
0	4	2,4025	4,81	0,5	480,5	DS-3
0	8	5,9000	11,80	0,3	1180	OC-12
0	9	4,5070	9,01	0,5	901,4	DS-3
0	10	14,7600	29,52	0,3	2952	OC-12
1	8	1,7625	3,53	0,5	352,5	DS-3
2	0	5,6725	11,35	0,3	1134,5	OC-12
2	5	1,7635	3,53	0,5	352,7	DS-3
2	6	3,6350	7,27	0,5	727	DS-3
2	10	9,2750	18,55	0,3	1855	OC-12
2	14	6,2900	12,58	0,3	1258	OC-12
3	6	4,4255	8,85	0,5	885,1	DS-3
3	8	12,8585	25,72	0,3	2571,7	OC-12
3	12	7,4300	14,86	0,3	1486	OC-12
3	13	7,9585	15,92	0,5	1591,7	DS-3
3	14	7,2235	14,45	0,3	1444,7	OC-12
4	0	2,4025	4,81	0,5	480,5	DS-3
4	15	1,9530	3,91	0,5	390,6	DS-3
5	2	1,7635	3,53	0,5	352,7	DS-3
5	9	7,5565	15,11	0,5	1511,3	DS-3
6	2	3,6350	7,27	0,5	727	DS-3
6	3	4,4255	8,85	0,5	885,1	DS-3
6	14	3,1785	6,36	0,5	635,7	DS-3
7	10	0,4440	0,89	0,3	88,8	OC-12
7	12	2,7800	5,56	0,5	556	DS-3
8	0	5,9000	11,80	0,3	1180	OC-12
8	1	1,7625	3,53	0,5	352,5	DS-3
8	3	12,8585	25,72	0,3	2571,7	OC-12
8	14	4,6890	9,38	0,3	937,8	OC-12
9	0	4,5070	9,01	0,5	901,4	DS-3
9	5	7,5565	15,11	0,5	1511,3	DS-3
10	0	14,7600	29,52	0,3	2952	OC-12
10	2	9,2750	18,55	0,3	1855	OC-12
10	7	0,4440	0,89	0,3	88,8	OC-12
10	12	3,0410	6,08	0,3	608,2	OC-12
11	12	0,6035	1,21	0,5	120,7	DS-3
12	3	7,4300	14,86	0,3	1486	OC-12
12	7	2,7800	5,56	0,5	556	DS-3
12	10	3,0410	6,08	0,3	608,2	OC-12
12	11	0,6035	1,21	0,5	120,7	DS-3

12	13	5,4625	10,93	0,3	1092,5	OC-12
13	3	7,9585	15,92	0,5	1591,7	DS-3
13	12	5,4625	10,93	0,3	1092,5	OC-12
14	3	7,2235	14,45	0,3	1444,7	OC-12
14	6	3,1785	6,36	0,5	635,7	DS-3
14	8	4,6890	9,38	0,3	937,8	OC-12
15	4	1,9530	3,91	0,5	390,6	DS-3
14	2	6,2900	12,58	0,3	1258	OC-12

Tabla 6 MCI – Matriz de la red

4.2.3 Topología Sprint

Ciudades	Nodo #	Potencia (mW)	Potencia (dBm)
Anaheim	0	1,56	1,9
Atlanta	1	2,5	4
Cheyenne	2	4,8	6,8
Chicago	3	3	4,8
Fort Worth	4	8	9
Kansas City	5	1,6	2
New York	6	6,7	8,3
Orlando	7	3,4	5,3
Pennsauken	8	6	7,8
Raleigh	9	9,1	9,6
Relay	10	8	9
Roachdale	11	5,7	7,6
San José	12	3,2	5,1
Seattle	13	10	10
Springfield	14	2	3
Stockton	15	1,1	0,4
Tacoma	16	4,6	6,6

Tabla 7 Sprint – Nodos Topologías Potencia de Láser

Porcentaje	Origen	Destinos				
30%	1	0	2	7	15	16
90%	1	4	7	10	12	13
	5	0	8	9	13	16
	10	1	8	12	14	15

Tabla 8 Sprint – Nodos Destinos

Nodo_i	Nodo_j	Retardo (ms)	Distancia*	Atenuacion (dB)	Distancia (Km)
0	4	9,4710	18,94	0,30	1894,2000
0	12	2,4175	4,84	0,30	483,5000
1	3	4,6250	9,25	0,30	925,0000
1	4	5,9130	11,83	0,30	1182,6000
1	6	5,9550	11,91	0,30	1191,0000
1	7	3,1930	6,39	0,30	638,6000
1	9	2,7830	5,57	0,30	556,6000
2	3	6,9980	14,00	0,30	1399,6000
2	16	7,5725	15,15	0,30	1514,5000
3	1	4,6250	9,25	0,30	925,0000
3	2	6,9980	14,00	0,30	1399,6000
3	4	6,4585	12,92	0,30	1291,7000
3	5	3,2455	6,49	0,30	649,1000
3	6	5,6280	11,26	0,30	1125,6000
3	11	1,9880	3,98	0,30	397,6000
3	12	14,4435	28,89	0,30	2888,7000
3	13	13,6160	27,23	0,30	2723,2000
3	14	6,0655	12,13	0,30	1213,1000
3	15	13,9960	27,99	0,30	2799,2000
4	0	9,4710	18,94	0,30	1894,2000
4	3	6,4585	12,92	0,30	1291,7000
4	5	3,6820	7,36	0,30	736,4000
4	7	7,7775	15,56	0,30	1555,5000
4	8	10,0350	20,07	0,30	2007,0000
5	3	3,2455	6,49	0,30	649,1000
5	4	3,6820	7,36	0,30	736,4000
5	8	8,0465	16,09	0,30	1609,3000
5	12	11,6300	23,26	0,30	2326,0000
5	13	11,8000	23,60	0,30	2360,0000
5	15	11,2435	22,49	0,30	2248,7000
6	1	5,9550	11,91	0,30	1191,0000
6	3	5,6280	11,26	0,30	1125,6000

6	8	1,5030	3,01	0,30	300,6000
6	10	2,3135	4,63	0,30	462,7000
6	14	0,9010	1,80	0,30	180,2000
7	1	3,1930	6,39	0,30	638,6000
7	4	7,7775	15,56	0,30	1555,5000
8	4	10,0350	20,07	0,30	2007,0000
8	5	8,0465	16,09	0,30	1609,3000
8	6	1,5030	3,01	0,30	300,6000
8	10	0,9170	1,83	0,30	183,4000
8	11	3,7475	7,50	0,30	749,5000
8	15	19,6480	39,30	0,30	3929,6000
9	1	2,7830	5,57	0,30	556,6000
9	10	1,1205	2,24	0,30	224,1000
10	6	2,3135	4,63	0,30	462,7000
10	8	0,9170	1,83	0,30	183,4000
10	9	1,1205	2,24	0,30	224,1000
10	12	18,9680	37,94	0,30	3793,6000
11	3	1,9880	3,98	0,30	397,6000
11	8	3,7475	7,50	0,30	749,5000
12	0	2,4175	4,84	0,30	483,5000
12	3	14,4435	28,89	0,30	2888,7000
12	5	11,6300	23,26	0,30	2326,0000
12	10	18,9680	37,94	0,30	3793,6000
12	15	0,4180	0,84	0,30	83,6000
12	16	5,5075	11,02	0,30	1101,5000
13	3	13,6160	27,23	0,30	2723,2000
13	5	11,8000	23,60	0,30	2360,0000
13	16	0,1950	0,39	0,30	39,0000
14	3	6,0655	12,13	0,30	1213,1000
14	6	0,9010	1,80	0,30	180,2000
15	3	13,9960	27,99	0,30	2799,2000
15	5	11,2435	22,49	0,30	2248,7000
15	8	19,6480	39,30	0,30	3929,6000
15	12	0,4180	0,84	0,30	83,6000
16	2	7,5725	15,15	0,30	1514,5000
16	12	5,5075	11,02	0,30	1101,5000
16	13	0,1950	0,39	0,30	39,0000

Tabla 9 Sprint – Matriz de la Red

4.3 EQUIPO COMPUTACIONAL

El experimento se está desarrollando sobre un computador con las siguientes características:

- Procesador Pentium 4 de 2.8GHz
- Memoria RAM de 512 MB
- Windows XP.
- Lenguaje Java Version 4.1

4.4 COMPARACIÓN DE RESULTADOS

A continuación se muestran los valores promedios arrojados por las 100 corridas en las 5 ejecuciones de los algoritmos MOANTCOL y MOANTCOLE al ser aplicados en las topologías NSF, MCI y SPRINT, tomando el 30% y 90% de sus nodos como nodos destinos:.

ALGORITMO MOANTCOL

NFS 30	f_1	f_2	F_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}
1	98.22	16.25	14.75	7.375	6.125	3.88	3.63	37.04	24.27	15.56	4.32
2	98.22	16.25	14.75	7.375	6.125	3.88	3.63	37.04	24.27	15.56	4.32
3	98.22	16.25	14.75	7.375	6.125	3.88	3.63	37.04	24.27	15.56	4.32
4	103.04	16.57	14.86	7.43	6.29	4	3.71	39.00	25.15	14.76	4.20
5	98.22	16.25	14.75	7.375	6.125	3.88	3.63	37.04	24.27	15.56	4.32

Tabla 10

ALGORITMO MOANTCOLE

NFS 30	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}
1	81.48	14.26	20.16	10.08	4.74	2.2	3.14	26.65	13.48	13.37	4.68
2	85.81	14.74	21.46	10.73	4.81	2.06	3.26	27.97	13.11	13.45	4.65

3	80.19	14.22	20.08	10.04	4.73	2.22	3.14	26.24	13.50	13.05	4.69
4	73.20	13.47	19.16	9.58	4.47	2.05	2.98	25.17	13.74	13.30	4.55
5	75.27	13.52	19.45	9.73	4.5	2.07	2.98	25.65	13.61	13.82	4.51

Tabla 11

ALGORITMO MOANTCOL

NFS 90	f_1	f_2	F_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}
1	214.96	43.33	44.93	22.47	6.67	5.67	3	35.34	32.04	16.54	5.70
2	235.36	44.58	45.67	22.83	6.67	5.67	3	39.32	36.02	18.10	5.56
3	198.22	42.44	43.33	21.67	6.00	5.00	2.89	29.83	26.53	15.25	5.58
4	231.83	43.33	44.89	22.44	6.78	5.78	3	41.85	38.55	17.83	5.39
5	225.17	43.27	44.73	22.36	6.36	5.36	3	38.88	35.58	17.32	5.74

Tabla 12

ALGORITMO MOANTCOLE

NFS 90 EV	f_1	F_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}
1	167.73	35.31	43.87	21.93	5.07	4.07	2.16	25.90	22.60	12.90	5.58
2	172.27	34.00	42.98	21.49	4.47	3.47	2.09	25.93	22.63	13.25	5.53
3	173.03	34.24	42.76	21.38	4.53	3.53	2.16	25.87	22.57	13.31	5.53
4	168.42	35.29	44.00	22.00	4.94	3.94	2.18	26.10	22.80	12.96	5.54
5	172.73	34.35	43.18	21.59	4.57	3.57	2.18	26.00	22.70	13.29	5.50

Tabla 13

ALGORITMO MOANTCOL

MCI 30	f_1	f_2	F_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}
1	47.61	9	12	6	3.5	2.5	1	17.92	14.29	8.52	4.26
2	47.61	9	12	6	3.5	2.5	1	17.92	14.29	8.52	4.26
3	47.61	9	12	6	3.5	2.5	1	17.92	14.29	8.52	4.26
4	47.61	9	12	6	3.5	2.5	1	17.92	14.29	8.52	4.26
5	47.61	9	12	6	3.5	2.5	1	17.92	14.29	8.52	4.26

Tabla 14

ALGORITMO MOANTCOLE

MCI 30 EV	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}
1	40.18	8	12	6	3	2	1	12.92	9.28	7.04	4.26
2	40.18	8	12	6	3	2	1	12.92	9.28	7.04	4.26
3	40.18	8	12	6	3	2	1	12.92	9.28	7.04	4.26
4	40.18	8	12	6	3	2	1	12.92	9.28	7.04	4.26
5	40.18	8	12	6	3	2	1	12.92	9.28	7.04	4.26

Tabla 15

ALGORITMO MOANTCOL

MCI 90	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}
1	199.480	42.000	39.600	19.800	5.000	4.000	2.400	29.263	27.501	14.249	5.475
2	208.149	42.250	39.500	19.750	5.000	4.000	2.500	30.889	29.127	14.868	5.428
3	201.662	42.000	39.200	19.600	5.000	4.000	2.400	29.358	27.596	14.404	5.475
4	197.062	41.833	39.333	19.667	5.000	4.000	2.333	28.259	26.496	14.076	5.506
5	197.062	41.833	39.333	19.667	5.000	4.000	2.333	28.259	26.496	14.076	5.506

Tabla 16

ALGORITMO MOANTCOLE

MCI 90 EV	f_1	F_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}
1	204.43	42.50	40.50	20.25	5.00	4.00	2.50	30.03	28.26	14.60	5.43
2	196.76	41.83	39.33	19.67	5.00	4.00	2.33	28.26	26.50	14.05	5.51
3	204.43	42.50	40.50	20.25	5.00	4.00	2.50	30.03	28.26	14.60	5.43
4	196.76	41.83	39.33	19.67	5.00	4.00	2.33	28.26	26.50	14.05	5.51
5	204.50	42.50	40.50	20.25	5.00	4.00	2.50	30.03	28.26	14.61	5.43

Tabla 17

ALGORITMO MOANTCOL

SPRINT30	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}
1	97.42	13.00	15.33	7.67	4.00	3.00	2.33	30.69	27.50	12.48	5.32
2	97.42	13.00	15.33	7.67	4.00	3.00	2.33	30.69	27.50	12.48	5.32
3	97.42	13.00	15.33	7.67	4.00	3.00	2.33	30.69	27.50	12.48	5.32
4	97.42	13.00	15.33	7.67	4.00	3.00	2.33	30.69	27.50	12.48	5.32
5	97.42	13.00	15.33	7.67	4.00	3.00	2.33	30.69	27.50	12.48	5.32

Tabla 18

ALGORITMO MOANTCOLE

SPRINT30 EV	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}
1	67.64	10.00	15.00	7.50	3.00	2.00	2.00	18.91	15.72	9.53	6.15
2	67.64	10.00	15.00	7.50	3.00	2.00	2.00	18.91	15.72	9.53	6.15
3	67.64	10.00	15.00	7.50	3.00	2.00	2.00	18.91	15.72	9.53	6.15
4	67.63	10.67	15.33	7.67	3.33	2.33	2.00	19.00	15.81	9.53	5.71
5	67.44	11.00	16.00	8.00	3.50	2.50	2.00	18.86	15.67	9.49	6.71

Tabla 19

ALGORITMO MOANTCOL

SPRINT30	f_1	F_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}
1	204.34	29.91	43.09	21.55	4.27	3.27	1.64	32.05	31.14	13.62	8.75
2	219.69	32.12	43.65	21.82	4.65	3.65	1.94	35.19	34.27	14.65	8.92
3	200.85	31.25	45.13	22.56	4.44	3.44	1.81	30.87	29.95	13.39	8.44
4	202.33	31.50	44.60	22.30	4.60	3.60	1.80	30.96	30.04	13.49	8.38
5	213.70	31.67	44.44	22.22	4.44	3.44	1.78	31.82	30.90	14.25	8.77

Tabla 20

ALGORITMO MOANTCOLE

SPRINT30 EV	f_1	F_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}
1	163.07	25.47	42.27	21.13	3.00	2.00	1.00	21.87	20.96	10.87	8.51
2	170.47	25.29	40.57	20.29	3.00	2.00	1.00	24.71	23.80	11.36	8.70
3	166.15	25.20	41.00	20.50	3.00	2.00	1.00	23.12	22.20	11.08	8.56
4	163.01	25.47	42.27	21.13	3.00	2.00	1.00	21.87	20.96	10.87	8.51
5	166.15	25.20	41.00	20.50	3.00	2.00	1.00	23.12	22.20	11.08	8.56

Tabla 21

Partiendo de la premisa de que el algoritmo MOANTCOL, es un algoritmo comprobado, se asume como una buena aproximación los valores arrojados por este algoritmo. Por otra parte, en nuestro problema, el objetivo es minimizar cada una de las 11 funciones objetivos que constituyen el modelo. En estos resultados, se observa para cada una de las topologías, que los resultados obtenidos de los promedios de las 100 corridas durante las 5 ejecuciones del algoritmo MOANTCOLE, dominan en algunos casos débilmente y en otros fuertemente

a los resultados producidos por el algoritmo MOANTCOL. El hecho anterior nos da bases para afirmar que el algoritmo MOANTCOLE esta convergiendo a la solución verdadera, aunque desconocida, con mejor aproximación y esto se confirmará mediante el análisis funcional de las soluciones obtenidas a través de la aplicación de las siguientes métricas: *Generación de vectores no dominados* (GVND), la *Distancia generacional* (DG) y el *Spacing* (S).

Por otro lado, como no se conoce el frente de Pareto real o true, este será asociado con el frente de Pareto producido por el algoritmo MOANTCOLE y el frente de Pareto aproximado con el frente de Pareto producido por el algoritmo MOANTCOL.

4.4.1 Generación de vectores no dominados (GVND). La métrica GVND, índica el número de elementos no dominados generados por el algoritmo, es decir nos muestra cuántos elementos hay en el frente de Pareto.

$$GVND = \left| PF_{true} \right| \qquad \textbf{(1)}$$

Los resultados obtenidos para cada una de las topologías se muestran a continuación:

4.4.1.1 Resultados Topología con 30% de sus nodos como nodos destino. Cuando se tomó el 30% de los nodos de la red como nodos destinos, los datos obtenidos para el comportamiento de la Generación de Vectores No Dominados para cada una de las topologías se presentan en la tabla 22.

Topología	MOANTCOL		MOANTCOLE	
	Promedio	**Desv. Estándar**	**Promedio**	**Desv. Estándar**
NSF	7.8	0.4472	51.2	10.94
MCI	2	0	1	0
Sprint	3	0	2.6	0.89

Tabla 22 GVND – Cantidad Nodos destinos: 30%

Se observa que el algoritmo MOANTCOLE, presenta un promedio más elevado de soluciones no dominadas, que el algoritmo MOANTCOL. En cambio para la topología MCI, el algoritmo MOANTCOL obtuvo un promedio de dos soluciones no dominadas, mientras que el MOANTCOLE tuvo un promedio de una solución no dominada, debido a que el conjunto de soluciones fueron las mismas, o tal vez numéricamente muy cercanas (alrededor de alguna vecindad).

Para la topología Sprint, al ser aplicado los dos algoritmos, tuvo un comportamiento similar a la presentada por la topología MCI, con la diferencia de tener una mayor desviación estándar lo cual produce una ligera variabilidad en el conjunto de soluciones no dominadas obtenidas.

4.4.1.2 Resultados Topología con 90% de sus nodos como nodos destino. Cuando se tomó el 90% de los nodos de la red como nodos destinos, los datos obtenidos para el comportamiento de la Distancia Generacional para cada una de las topologías se presentan en la tabla 23.

Topología	MOANTCOL		MOANTCOLE	
	Promedio	**Desv. Estándar**	**Promedio**	**Desv. Estándar**
NSF	11.2	2.4899	47.4	3.2863
MCI	5.2	0.8366	7.2	1.0954
Sprint	11.4	3.5071	12.6	3.6469

Tabla 23 GVND – Cantidad Nodos destinos: 90%

Se observa que el algoritmo MOANTCOLE, muestra la misma tendencia que presentó cuando se tomaron el 30% de los nodos como nodos destinos, la de un promedio más elevado de soluciones no dominadas sobre las que presenta el algoritmo MOANTCOL. En cambio para las topologías MCI y Sprint, los algoritmos MOANTCOL y MOANTCOLE presentan una ligera mejoría en cuanto a la generación de vectores no dominados, teniendo el algoritmo MOANTCOLE, una pequeña ventaja numérica en cantidad y desviación estándar. Debido a los resultados obtenidos por los algoritmos en las

redes MCI y Sprint, podría inferirse que ha pesar de tener más nodos pero menos enlaces, este hecho posiblemente influya en que se obtengan una menor generación de vectores no dominados.

4.4.2 Distancias Generacionales (DG). La métrica DG, nos índica qué tan lejos está el frente de Pareto aproximado del Frente de Pareto real o true.

$$DG = \frac{\sqrt{\sum_{i=1}^{|PF_{known}|} d_i}}{|PF_{known}|} \qquad \textbf{(2)}$$

Donde d_i es la mínima distancia Euclidiana entre el i-ésimo elemento del frente de Pareto aproximado con los elementos del frente de Pareto real.

Los resultados y el análisis de los promedios obtenidos de las once funciones objetivo para cada una de las topologías se muestran a continuación.

4.4.2.1 Resultados Topología con 30% de sus nodos como nodos destino. Cuando se tomó el 30% de los nodos de la red como nodos destinos, los datos obtenidos para el comportamiento de la Distancia Generacional, a cada una de las topologías se presentan en la tabla 24:

Topología	DG	
	Promedio	**Desv. Estándar**
NSF	2.2453	0,0752
MCI	8.2135	0
Sprint	29.9964	0

Tabla 24 DG – Cantidad Nodos destinos: 30%

Se observa que el distanciamiento del frente de Pareto aproximado, en este caso, el producido por el algoritmo MOANTCOL, con respecto al frente de Pareto true (el

producido por el algoritmo MOANTCOLE), fue mucho mayor en la topología de 17 nodos Sprint, pero el distanciamiento también está presente para las topologías NSF y MCI.

4.4.2.2 Resultados Topología con 90% de sus nodos como nodos destino. Cuando se tomó el 90% de los nodos de la red como nodos destinos, los datos obtenidos para el comportamiento de la Distancia Generacional a cada una de las topologías se muestran a continuación en la tabla 25:

Topología	DG	
	Promedio	**Desv. Estándar**
NSF	2.0935	0,1036
MCI	0.5224	0,77459
Sprint	9,28	2.5663

Tabla 25 DG – Cantidad Nodos destinos: 90%

Se observa que el distanciamiento del frente de Pareto aproximado, en este caso, el producido por el algoritmo MOANTCOL, con respecto al frente de Pareto true (el producido por el algoritmo MOANTCOLE), conserva la misma tendencia: la de ser mucho mayor en la topología de Sprint, pero para la topología MCI se presenta proximidad entre los frentes de Pareto y además de una variación en la desviación estándar.

4.4.3 Spacing (S). La métrica S, nos índica qué tan bien están distribuidas las soluciones sobre el frente de Pareto.

$$S = \sqrt{\frac{n-1}{n}\left(\frac{\sum_{i=1}^{|PF_{known}|}(\bar{d}-d_i)}{|PF_{known}|-1}\right)} \qquad \textbf{(3)}$$

Donde $d_i = \min_j \left(\sum_{k=1}^{m} \left| f_k{}^i(x_i) - f_k{}^j(x_j) \right| \right)$ con $i \neq j$ y $1 \leq i, j \leq |PF_{known}|$, y $\overline{d}$ es el promedio de todos los d_i.

Para esta métrica un valor de 0 significa que todos lo elementos de PF_{known} están equidistantemente espaciados. Por lo tanto, entre menor sea el valor de *S*, mejor es la distribución de los elementos del PF_{known}.

Los resultados y el análisis de los promedios obtenidos de las once funciones objetivo para cada una de las topologías se muestran a continuación:

4.4.3.1 Resultados Topología con 30% de sus nodos como nodos destino. Cuando se tomó el 30% de los nodos de la red como nodos destinos, los datos obtenidos para el comportamiento del Spacing distribución del espaciamiento de las soluciones en cada una de las topologías se presentan en la tabla 26:

Topología	MOANTCOL		MOANTCOLE	
	Promedio	**Desv. Estándar**	**Promedio**	**Desv. Estándar**
NSF	1.0008	0.0337	0.3368	0.0585
MCI	0	0	0	0
Sprint	2.5149	1.5498	0.0337	0.0465

Tabla 26 Spacing – Cantidad Nodos destinos: 30%

Se observa que en las tres topologías el algoritmo MOANTCOLE, tiene una mejor distribución de sus soluciones que las producidas por el algoritmo MOANTCOL; además se obtiene una distribución de las soluciones completamente equilibrada en su espaciamiento, en la topología MCI.

4.4.3.2 Resultados Topología con 90% de sus nodos como nodos destino. Cuando se tomó el 90% de los nodos de la red como nodos destinos, los datos obtenidos para el comportamiento del Spacing distribución del espaciamiento de las soluciones en cada una de las topologías se presentan en la tabla 27:

Topología	MOANTCOL		MOANTCOLE	
	Promedio	**Desv. Estándar**	**Promedio**	**Desv. Estándar**
NSF	0.7755	0,2017	0.2667	0,0259
MCI	0.5410	0.5227	0.0019	0.0043
Sprint	0.8309	0.1728	0.3472	0.0587

Tabla 27 Spacing – Cantidad Nodos destinos: 90%

Se observa en las tres topologías que el algoritmo MOANTCOLE, tiene una mejor distribución de sus soluciones que las producidas por el algoritmo MOANTCOL; además se obtiene una distribución de las soluciones casi completamente equilibrada en su espaciamiento, en la topología MCI, luego permanece la tendencia observada en las topologías con el 30% de sus nodos como nodos destinos.

4.4.4 Tiempo Computacional

4.4.4.1 Resultados Topología con 30% de sus nodos como nodos destino. Cuando se tomó el 30% de los nodos de la red como nodos destinos, los datos obtenidos para el comportamiento del tiempo computacional en cada una de las topologías se presentan en la tabla 28:

Topología	MOANTCOL		MOANTCOLE	
	Promedio	**Desv. Estándar**	**Promedio**	**Desv. Estándar**
NSF	3.994	0.0082	27.4004	13.6420
MCI	4.7340	0.9782	8.9028	0.2910
Sprint	4.3154	0.0172	8.7972	0.0574

Tabla 28 Tiempo de Ejecución (s) – Cantidad Nodos destinos: 30%

Se observa un mayor consumo de tiempo computacional en el algoritmo MOANTCOLE, marcándose una mayor diferencia y variabilidad (una desviación estándar grande), en la topología de red NSF.

4.4.4.2 Resultados Topología con 90% de sus nodos como nodos destino. Cuando se tomó el 90% de los nodos de la red como nodos destinos, los datos obtenidos para el comportamiento del tiempo computacional en cada una de las topologías se presentan en la tabla 29:

Topología	MOANTCOL		MOANTCOLE	
	Promedio	**Desv. Estándar**	**Promedio**	**Desv. Estándar**
NSF	4.3594	0.0005	8.8158	0.5618
MCI	4.5334	0.0688	13.2686	0.3793
Sprint	4.8124	0.0452	9.3378	0.0387

Tabla 29 Tiempo de Ejecución (s) – Cantidad Nodos destinos: 90%

Se observa una disminución en el consumo de tiempo computacional en el algoritmo MOANTCOLE, cuando opera sobre la red NSF y un ligero aumento en el consumo de

tiempo para la red MCI. La tendencia de un mayor consumo de tiempo computacional para el algoritmo MOANTCOLE se mantiene.

A continuación presentamos un resumen estadístico de la prueba de hipótesis diferencia de medias realizada sobre los promedios de los resultados obtenidos:

Análisis estadístico de la diferenci a de medias (niv el de confianza 95%

Topología	Métrica	σ	Dif Medias	Interv. Conf
NSF 30	GVND	σ1≠σ2	û1≠û2	[-45.6828;-41.1172]
NFS 90	GVND	σ1≠σ2	û1≠û2	[-39.6972;-32.7028]
MCI 30	GVND	σ1= σ2	û1=û2	
MCI 90	GVND	σ1= σ2	û1≠û2	[-2.65224;-1.34776]
SPRINT 30	GVND	σ1≠σ2	û1≠û2	[0.164855;0.964855]
SPRINT 90	GVND	σ1= σ2	û1=û2	[-4.31097;1.91097]
NSF 30	Spacing	σ1= σ2	û1≠û2	[0.594324;0.733792]
NFS 90	Spacing	σ1≠σ2	û1≠û2	[0.259226;0.758426]
MCI 30	Spacing	σ1= σ2	û1=û2	
MCI 90	Spacing	σ1≠σ2	û1=û2	[-0.109966;1.18823]
SPRINT 30	Spacing	σ1≠σ2	û1≠û2	[0.55731;4.40519]
SPRINT 90	Spacing	σ1= σ2	û1≠û2	[0.295511;0.671987]
NSF 30	Tiempo	σ1≠σ2	û1≠û2	[-37.4751;-9.33769]
NFS 90	Tiempo	σ1≠σ2	û1≠û2	[-5.03582;-3.87698]
MCI 30	Tiempo	σ1≠σ2	û1≠û2	[-5.36483;-2.97277]
MCI 90	Tiempo	σ1≠σ2	û1≠û2	[-9.20146;-8.26694]
SPRINT 30	Tiempo	σ1≠σ2	û1≠û2	[-4.55209;-4.41151]
SPRINT 90	Tiempo	σ1= σ2	û1≠û2	[-4.58676;-4.46484]

Estos resultados estadísticos nos confirman el mejor desempeño del algoritmo MOANTCOLE sobre el MOANTCOL.

4.4.5 Análisis de Correlación. Con el fin de comprender la relación entre el comportamiento de cada una de las funciones objetivo, se analiza el coeficiente de correlación, teniendo en cuenta que el coeficiente de correlación tiene valores adimensionales que oscilan entre -1 y 1. El coeficiente de correlación tomará un valor de 1 cuando la relación entre las variables es perfecta y directamente proporcional, es decir, hay dependencia lineal entre ellas. El coeficiente tomará un valor de -1 cuando la relación

entre las variables a analizar es perfecta e inversamente proporcional, es decir, hay independencia lineal entre ellas.

Para realizar el análisis de correlación se tomó la topología NSF con la cantidad de nodos destinos correspondientes al 30%, y 90%; los coeficientes de correlación obtenidos se encuentran consignados en las tablas 30 y 31; en las cuales se observa una estrecha relación entre varias funciones objetivo; sin embargo las relaciones que prevalecen son las siguientes:

CORRELACION											
	f_1	f_2	f_3	F_4	F_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}
f_1		0.9912	0.9678	0.9678	0.9709	0.2991	0.9844	0.9774	-0.9507	1	0.7766
f_2	0.9912		0.9575	0.9575	0.982	0.3326	0.9987	0.9568	-0.9296	0.9912	0.8266
f_3	0.9678	0.9575		1	0.8926	0.0652	0.9571	0.9932	-0.9951	0.9678	0.6302
f_4	0.9678	0.9575	1		0.8926	0.0652	0.9571	0.9932	-0.9951	0.9678	0.6302
f_5	0.9709	0.982	0.8926	0.8926		0.5026	0.977	0.8997	-0.8581	0.9709	0.9022
f_6	0.2991	0.3326	0.0652	0.0652	0.5026		0.3192	0.0931	-0.0091	0.2991	0.7626
f_7	0.9844	0.9987	0.9571	0.9571	0.977	0.3192		0.9511	-0.9272	0.9844	0.8268
f_8	0.9774	0.9568	0.9932	0.9932	0.8997	0.0931	0.9511		-0.9899	0.9774	0.634
f_9	-0.9507	-0.9296	-0.9951	-0.9951	-0.8581	-0.091	-0.9272	-0.9899		-0.9507	-0.5636
F_{10}	1	0.9912	0.9678	0.9678	0.9709	0.2991	0.9844	0.9774	-0.9507		0.7766
F_{11}	0.7766	0.8266	0.6302	0.6302	0.9022	0.7626	0.8268	0.634	-0.5636	0.7766	

Tabla 30 Coeficientes de Correlación de Funciones Objetivo – Topología NSF, destinos 30%

Observamos que la matriz de correlación conformada por los promedios producidos por cada una de las once funciones objetivo calculadas para la topología NSF tomando como nodos destinos el 30% de sus nodos, tiene una alta correlación entre todas sus funciones, la única función que resultó independiente fue la de retardo promedio (***f*₆**).

CORRELACION											
	***f*₁**	***f*₂**	***f*₃**	***f*₄**	***f*₅**	***f*₆**	***f*₇**	***f*₈**	***f*₉**	***f*₁₀**	***f*₁₁**
***f*₁**		-0.9526	-0.953	-0.953	-0.9723	-0.9723	-0.2907	-0.3602	-0.3602	1	-0.7913
***f*₂**	-0.9526		0.9523	0.9523	0.9855	0.9855	0.5654	0.4399	0.4399	-0.9526	0.6628
***f*₃**	-0.953	0.9523		1	0.9341	0.9341	0.4401	0.5984	0.584	-0.953	0.576
***f*₄**	-0.953	0.9523	1		0.9341	0.9341	0.4401	0.5948	0.5984	-0.953	0.576
***f*₅**	-0.9723	0.9855	0.9341	0.9341		1	0.4689	0.3103	0.3103	-0.9723	0.7682
***f*₆**	-0.9723	0.9855	0.9341	0.9341	1		0.4689	0.3103	0.3103	-0.9723	0.7682
***f*₇**	-0.2907	0.5654	0.4401	0.4401	0.4689	0.4689		0.4449	0.4449	-0.2907	-0.0856
***f*₈**	-0.3602	0.4399	0.5984	0.5984	0.3103	0.3103	0.4449		1	-0.3602	-0.2732
***f*₉**	-0.3602	0.4399	0.5984	0.5984	0.3103	0.3103	0.4449	1		-0.3602	-0.2732
***f*₁₀**	1	-0.9526	-0.953	-0.953	-0.9723	-0.9723	-0.2907	-0.3602	-0.3602		-0.7913
***f*₁₁**	-0.7913	0.6628	0.576	0.576	0.7682	0.7682	-0.0856	-0.2731	-0.2732	-0.7913	

Tabla 31 Coeficientes de Correlación de Funciones Objetivo – Topología NSF, destinos 90%

Se observa la misma tendencia que en el caso anterior, para la matriz de correlación conformada por los promedios producidos por cada una de las once funciones objetivos, para la topología NSF al tomar como nodos destinos 90% de sus nodos, la de tener una alta

correlación entre todas sus funciones, tan solo dos funciones resultaron independientes: retardo promedio (***f_6***) *y la atenuación* (***f_{11}***).

La alta correlación de los datos nos permite inferir que el modelo matemático puede ser expresado con menos funciones, esto es una consecuencia de que funciones como retardo promedio, retardo máximo y variación máxima del retardo dependan de la función de retardo total y saltos promedios, saltos máximos y variación máxima de saltos dependan de la función de salto total. Por otra parte, es posible que en el diseño de la red existan valores tales, en que a pesar de que funciones matemáticamente independientes tales como retardo y saltos se presente la posibilidad de que a mas saltos más retardos ó viceversa lo cual induce a una correlación.

4.4.6 Intervalo de Confianza. Este estimado busca encontrar el intervalo en el que varía un valor de una población, a partir del promedio y la desviación estándar del mismo. Con el fin de analizar este estimado, se tomó la topología NSF para 30% y 90% de sus nodos como nodos destinos, en las tablas 32 y 33 se consignan los intervalos de los promedios de cada una de las funciones objetivo, tanto para el algoritmo MOANTCOL, como el MOANTCOLE:

Funciones	**Intervalo MOANTCOL**		**Intervalo MOANTCOLE**	
f_1	202.52	239.694	167.671	173.997
f_2	42.4441	44.3428	33.8734	35.4076
f_3	43.654	45.7658	42.672	44.0386
f_4	21.827	22.8829	21.336	22.0193
f_5	6.1017	6.8888	4.3804	5.0501
f_6	5.1017	5.8881	3.3804	4.0501
f_7	2.9160	3.0394	2.1058	2.1953
f_8	31.2671	42.8242	25.8454	26.076
f_9	27.9671	39.5242	22.5454	22.776
f_{10}	15.5785	18.438	12.8977	13.3844
f_{11}	5.4225	5.7662	5.5023	5.5716

Tabla 32 Intervalos de Confianza 95% – Topología NSF, destinos 30%

Funciones	Intervalo MOANTCOL		Intervalo MOANTCOLE	
F_1	202.52	239.96	167.67	174.00
F_2	42.44	44.34	33.87	35.41
F_3	43.65	45.77	42.67	44.03
F_4	21.83	22.88	21.34	22.10
F_5	6.10	6.89	4.38	5.05
F_6	5.10	5.89	3.38	4.05
F_7	2.91	3.04	2.11	2.20
F_8	31.27	42.82	25.84	26.08
F_9	27.97	39.52	22.55	22.78
F_{10}	15.58	18.44	12.90	13.38
F_{11}	5.42	5.77	5.50	5.57

Tabla 33 Intervalos de Confianza 95% – Topología NSF, destinos 90%

En ambos casos se evidencian intervalos de confianza con límites inferior y superior menores, producidos por el algoritmo MOANTCOLE, lo cual es un indicativo de la dominancia a nivel de promedios por parte de este algoritmo.

4.4.7 Caso de prueba. Todos los resultados obtenidos anteriormente, hacen necesaria la verificación de la cantidad de generaciones calculadas, esperando con ello encontrar mejores resultados o la de verificar tendencias similares. Para este fin, se tomó la topología NSF, con una cantidad de nodos destinos correspondientes al 30% y 90 % de sus nodos y se aplicaron los algoritmos MOANTCOL y MOANTCOLE bajo las mismas condiciones del experimento anterior, exceptuando el número de generaciones que sufrió una variación del doble al tomar 200 (en el experimento anterior se tomaron 100). A continuación mostramos los resultados obtenidos.

4.4.7.1 Generación de vectores no dominados (GVND)

Destinos	MOANTCOL		MOANTCOLE	
	Promedio	Desv. Estándar	Promedio	Desv. Estándar
30%	8	0	54.2	3,1947
90%	14	0.7071	53.4	0.5477

Tabla 34 GVND – Topología NSF – 150 Generaciones

Para la métrica GVND, se observa que el promedio de vectores no dominados sigue siendo mayor en el algoritmo MOANTCOLE, además se presenta un incremento de vectores no dominados en la red con 30% de sus nodos como nodos destinos del 30% y para la red con el 90% de sus nodos como nodos destinos, un incremento de 20%, por parte del algoritmo MOANTCOL (al comparar los valores de la tabla 22 con los valores de la tabla 11). Los incrementos en la Generación de vectores no dominados, al tomar en la topología NSF el 30% y 90% de sus nodos como nodos destinos, cuando se aplicó el algoritmo MOANTCOLE, mostraron un incremento de 5.53% y 11.23% respectivamente (al comparar los valores de la tabla 34 con los valores de la tabla 11).

4.4.7.2 Distancia Generacionales (DG)

Destinos	DG	
	Promedio	Desv. Estándar
30%	7.2799	0
90%	7.2003	2.5990

Tabla 35 DG – Topología NSF– 150 Generaciones

La distancia generacional promedio producida al tomar como frente de Pareto analítico las producidas por el algoritmo MOANTCOLE y como aproximadas las producidas por el algoritmo MOANTCOLE, muestra un incremento en su distanciamiento del 69:16% para la topología NSF con 30% de sus nodos destinos y del 70.92% al tomar la red NSF con el

90% de sus nodos destinos, lo cual implica un mejoramiento en la calidad de las soluciones que se encuentran en el frente de Pareto producido por el algoritmo MOANTCOLE. Ver tablas 35, 24 y 25.

Spacing (S)

Destinos	MOANTCOL		MOANTCOLE	
	Promedio	**Desv. Estándar**	**Promedio**	**Desv. Estándar**
30%	0.9857	0	0.3249	0.0215
90%	0.8335	0.14	0.2089	0.0134

Tabla 36 Spacing – Topología NSF– 150 Generaciones

Al comparar los resultados obtenidos entre las tablas 36, 26 y 27 se observa un ligero mejoramiento en el Spacing promedio producido por el algoritmo MOANTCOLE al disminuir levemente para las topologías NSF con el 30% y 90% de sus nodos como nodos destinos. En cambio el algoritmo MOANTCOL tuvo una ligera mejoría en la topología con el 30% y un leve decremento con el 90% de sus nodos como nodos destinos.

4.4.7.3 Tiempo Computacional. Los tiempos computacionales de este nuevo experimento son visiblemente mayores, lo cual no es más que una consecuencia directa del aumento en la cantidad de generaciones a calcular (Tabla 37).

Destinos	MOANTCOL		MOANTCOLE	
	Promedio	**Desv. Estándar**	**Promedio**	**Desv. Estándar**
30%	19.3092	0.0303	251.0468	65.82
90%	22,2938	0.0202	56.4028	2.3916

Tabla 37 Tiempo Computacional (s) – Topología NSF – 150 Generaciones

5 CONCLUSIÓN Y TRABAJOS FUTUROS

5.1 CONCLUSIONES

En la presente investigación se desarrolló una solución computacional a la transmisión multicast en árboles origen–destino estáticos, sobre redes GMPLS, en las que se desean optimizar once (11) funciones objetivo y cuya solución se alcanza a través de metaheurísticas, presentando dos algoritmos Ant Colony (MOANTCOL) y Ant Colony Evolutivo (MOANTCOLE). Éste último es una propuesta que integra los procedimientos evolutivos del algoritmo SPEA2 en el Ant Colony.

De acuerdo con los resultados discutidos en el ítem 4.4, en los que se muestra el comportamiento de los algoritmos MOANTCOL y MOANTCOLE al ser probados sobre las topologías MCI, NSF y Sprint, se puede concluir:

- El algoritmo MOANTCOLE tuvo mejores resultados sobre el algoritmo MOANTCOL, al aplicar las métricas: Generación de Vectores No Dominados, Distancia Generacional y Spacing. Cuando se probaron los algoritmos antes mencionados sobre las topologías NSF, MCI y Sprint, con el 30 y 90% de sus nodos como nodos destinos; el algoritmo MOANTCOLE produjo mejores soluciones (valores en promedio más bajos); mayor cantidad de vectores no dominados, especialmente en la topología NSF, debido a que tiene más enlaces, y soluciones distribuidas o espaciadas más uniformemente (Spacing).

- El tiempo consumido por el algoritmo MOANTCOLE, siempre fue mayor debido a que el procedimiento de búsqueda local con cruce de caminos eleva la complejidad del algoritmo.

5.2 TRABAJOS FUTUROS

- Implementar el algoritmo Ant Colony, con estrategias de algoritmos evolutivos, tales como el NSGA II y el algoritmo memético PAES.

- Proponer el algoritmo en la solución de un modelo MOPs en transmisión Multicast dinámica, es decir un modelo en donde los nodos fuentes puedan salir y entrar, es decir los nodos fuentes puedan cambiar durante la conexión.

6 BIBLIOGRAFIA

ALMEROTH, Kevin. The Evolution of Multicast: From MBone to Inter-Domain Multicast to Internet2 deployment. Department of Computer Science, Unversity of California at Santa Bárbara. 1999.

ALVARADO, Carolina y HERAZO Iván. Análisis comparativo de los algoritmos evolutivos NSGA-II y SPEA-II mediante la solución de un problema de optimización multiobjetivo. Barranquilla, 2004, 228 p. Trabajo de grado (Ingeniero de sistemas). Universidad del Norte. División de Ingenierías. Departamento Ingeniería de sistemas.

BAASE, Sara and VAN GELDER, Allen Computer Algorithms: Introduction to design and Analysis. Massachusetts U.S.A. Addison Wesley.2002

BLUM, C., ROLI, A., and DORIGO, M. HC-ACO: The Hyper-Cube Framework for Ant Colony Optimization. IRIDIA-Universit´e Libre de Bruxelles.

CARO, Luis y ROSADO, Pierre. Optimización multi-objetivo en transmisiones multicast sobre redes ópticas usando algoritmos evolutivos, meméticos y ant-colony. Barranquilla, 2005, 203 p. Trabajo de grado (Ingeniero de sistemas). Universidad del Norte. División de Ingenierías. Departamento Ingeniería de sistemas.

COELLO, Carlos; Van Veldhuizen, David and LAMONT, Gary. Evolutionary Algorithms for Solving Multi-Objective Problems. México D.C. Kluwer Academics, 2002.

COLLETTE, Yann and SIARRY Patrick. Multiobjective Optimization Principles and Cases Study. Berlin:Springer-Verlag 2003.

CORONELL, Margarita y TOVAR, Luis, Optimización Multiobjetivo sobre Redes Ópticas en transmisión Multicast. Barranquilla 2006, 126 p. Trabajo de Grado (Maestría en Ciencias de la Computación). Instituto Tecnológico y de Estudios Superiores de Monterrey- Universidad Autónoma de Bucaramanga - Universidad Tecnológica de Bolívar
CHERIYAN, J and RAVI, R. Approximation algorithms for network problems. 1998, 222 p.

CHERIYAN, J and RAVI, R. Approximation algorithms for network problems. 1998, 222 p. Disponible electrónicamente en http://www.math.uwaterloo.ca/~jcheriya/PS_files/ln-master.ps, p 84.

DEB, Kalyanmoy and GOELAND, Tuschar. Hybrid Methods for Multi-Objective Evolutionary Algorithms. Kanpur Genetic Algorithms Laboratory (KanGAL). Indian Instituty of Technology, Kampur, India. 2002.

DONOSO, Yesid. Multi-objective Optimization Scheme for static and Dynamic Multicast Flow. Girona (España), 2005, 177 p. Trabajo de grado (Tesis Doctoral). Universidad de Girona. Departamento de Electrónica y Control automático.

DONOSO, Yezid; FABREGAT, Ramón y MARZO, José L. Multi-Objective Scheme over Multi-Tree Routing in Multicast MPLS Networks. En: Latin American Networking Conference (LANC'03). (2003 : La Paz).

DONOSO Y., FABREGAT R., SOLANO F., MARZO, J. GMM-model for Dynamic Multicast Groups using a probabilistic BFS algorithm". To be presented in IEEE ICC 2005. Seoul, Korea.

DONOSO, Yesid, PEREZ, Alfredo, ARDILA, Carlos. Optimizing Multiple Objectives on Multicast Networks using Memetic Algorithms. GETS International transaction on Computer Science and Engineering. Seul, Korea. October 2005.

DORIGO, Marco and STTZLE, Thomas. Ant Colony Optimization. Belgium. Bradford Books, 2004. 328p. ISBN 0262042193**.**

DORIGO, M and GAMBARDELLA, L. Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem. IEEE Transaction on Evolutionary Computation, Vol. 1. No. 1, 1997.

GALLO, Michael y HACOCK, William. Comunicación Entre Computadores y Tecnología de Redes. México D.C.: Thomson, 2002.

ISHIBUSHI, Hisao y YOSHIDA, Tadashi. Hybrid Evolutionary Muti-Objective Optimization Algorithms. Department of Industrial Engineering, Osaka Prefecture University. Osaka, Japan 2003.

JASKIEWICKZ, Andrzej. Genetic Local Search for Multiple Objective Combinatorial Optimization. Institute of Computing Science, Poznan University of Technology. Poznan, Poland 1998.

JASZKIEWICZ, Andrzej. Multiple objective metaheuristic algorithms for combinatorial optimization. Trabajo de Grado (Doctor Habilis, Computer Science), Poznan, 2001, 148p. Poznan University of Technology, Poznan, Poland. Institute of Computing Science

KNOWLES, Joshua. Local-Search and Hybrid Evolutionary Algorithms For Pareto Optimization. Reading, 2002, Trabajo de Grado (Doctor of Philosophy in Computer Science). University of Reading, England, Department of Computer Science.

KRASNOGOR, Natalio y GUSTAFSON, Steven. La Búsqueda Local como Proveedora de Bloques Constructivos en Algoritmos Meméticos Auto-Generados. Automated

Schedulling, Optimization and Planning Group. School of Computer Science and IT. University of Nottinngham, United Kindom. 2003.

MANIEZZO, Vittorio, GAMBARDELLA, Luca María y DE LUIGGI, Fabio. Ant Colony Optimization. Paper 2004.

MERZ, Peter. Memetic Algorithms for Combinatorial Optimization Problems. Siegen, German. Tesis de Doctorado. Universitat Gesamthochshule. 2000.

MICHALEWICZ, Zbigniew. Genetic Algorithms + Data Structures = Evolution Programs. Editorial Springer-Verlag. New York, 1994. p.53.

MILLER, C. Kenneth. Multicast Networking and Applications. United States: Addison Wesley, 1999. 282 p.

MOSCATO, Pablo. On Evolution, Search, Optimization, Genetic Algorithms and Martial Arts: Toward Memetic Algorithms. California Institute of Technology, Pasadena California USA, 1989.

MOSCATO, Pablo. Problemas de Otimização NP, Aproximabilidade e Computação Evolutiva: Da Prática à Teoria. Tese de Doutorado. Universidade Estadual de Campinas. Departamento de Engenharia de Sistemas. Facultade de Engenharia Elétrica e Computação. Campinas, Brasil, 2001.

MOSCATO, Pablo y COTTA, Carlos. Una Introducción a los Algoritmos Meméticos. Revista Iberoamericana de Inteligencia Artificial. No.19. 2003.

PEREZ, Alfredo. Optimización multiobjetivo en redes multicast mediante algoritmos , meméticos. Barranquilla, 2005, 203 p. Trabajo de grado (Ingeniero de sistemas). Universidad del Norte. División de Ingenierías. Departamento Ingeniería de sistemas.

SMITH, James Edward. Self Adaption in Evolutionary Algorithms. 112p.Trabajo de Grado (Doctor of Philosophy). Faculty of Computer Studies and Mathematics. University of the West England. Bristol, England, 1998.

STERN, Thomas y BALA Krishna. Multiwavelength Optical Networks a Layered Approach, New Jersey, USA : Prentice Hall 2000.

TANENBAUM, Adrew. Redes de computadores. Cuarta edición: Prentice Hall, 2003.

WITTMANN, R. and ZITTERBART M. Multicast Communication. Protocols and Applications. Morgan Kaufmann Publishers. 1999.

XU, Jiefeng, CHIU Steven and GLOVER, Freddy. A First Level Scattered Search Implementation for Solving the Steiner Ring Problem in Telecommunications Network Design. Aparecido en el libro Steiner Trees in Industries, Cheng X., Du, Z (ed), p. 441-466. Kluwer Academic Publishers. 2001.

ZIZTLER, Eckardt. Evolutionary Algorithms for Multiobjective Optimization: Methods and Issues. Swiss Federal Insitute of Technology. Zurich. Trabajo de Grado (Doctor of Technical Sciences). 232p. 1999

ZITZLER, E. and THIELE, L. An Evolutionary Algorithm for Multiobjective Optimization: The Strength Pareto Approach. Technical Report 43. Computer Engineering and Networks Laboratory (TIK). Swiss Federal Institute of Technology (ETH), Zurich, Switzerland, 1998.

ZITZLER, Eckart; LAUMANNS Marco and THIELE Lothar. SPEA2: Improving the Strength Pareto Evolutionary Algorithm. Report No. 103.Zurich: Swiss Federal Institute of Technology (ETH), 2000.

ZIZTLER, Eckardt. A Tutorial on Evolutionary Multiobjective Optimization. Computer Engineering and Networks Lab. Swiss Federal Institute of Technology (ETH) Zurich, Swiss.2000.